COMPOSITION CODES FOR MULTIPLE USER COMMUNICATION SYSTEMS

Timothy J. Healy

Department of Electrical Engineering and Computer Science

University of Santa Clara

CR-177353

March 30, 1985

(Supported by the National Aeronautics and Space Administration under Grant NCC2-128.)

COMPOSITION CODES FOR MULTIPLE USER COMMUNICATION SYSTEMS

Timothy J. Healy

Department of Electrical Engineering and Computer Science

University of Santa Clara

CR-177353

March 30, 1985

(Supported by the National Aeronautics and Space Administration under Grant NCC2-128.)

COMPOSITION CODES FOR MULTIPLE USER COMMUNICATION SYSTEMS

Timothy J. Healy

Department of Electrical Engineering and Computer Science

University of Santa Clara

March 30, 1985

(Supported by the National Aeronautics and Space Administration under Grant NCC2-128.)

COMPOSITION CODES FOR MULTIPLE USER COMMUNICATION SYSTEMS

ABSTRACT

Timothy J. Healy

Department of Electrical Engineering and Computer Science

University of Santa Clara

This report introduces and studies in detail a new algorithm which decodes code division multiple user communication signals. The algorithm makes use of the distinctive form or pattern of each signal to separate it from the composite signal created by the multiple users. Although the algorithm is presented in terms of frequency hopped signals, the actual transmitter modulator can use any of the existing digital modulation techniques.

The report also studies some of the codes which can be used in connection with the algorithm, and relates the algorithm to past studies which use other approaches to the same problem. Codes considered here include a set of deterministic codes, called "Prime Number Codes", which lead to zero inter-user interference, and random codes, in which the inter-user interference is finite but statistically controlled.

A number of techniques for the control of errors due to interference and due to noise are considered.

Finally, a practical application scenario is presented.

TABLE OF CONTENTS

COMPOSITION CODES FOR MULTIPLE USER COMMUNICATION SYSTEMS

1. INTRODUCTION

Many communication systems are designed to serve a number of users, often in situations in which the interconnections between users vary with time. Sometimes it may be necessary for a system to act in a "broadcast" mode in which one user transmits to all of the other users or to a subset of these users. At other times we may need to operate in a "multiple access" mode in which many users transmit to a single receiver. And of course a degenerate case of either of the above is the situation in which one user communicates with one other user. These various forms are referred to generically as "multiple user" commmunications.

In order to effect any of the above forms of communication, it is necessary to set up some form of channelization between the particular users which need to be interconnected. Given that the system has available some portion of the frequency-time-space spectrum [see, for example, Berry(1977)], it is possible to establish channels by dividing the spectrum in a number of ways, each of which is a form of "multiplexing". In each case it is necessary that the channelization provide that each user be as free as possible from interference from other users. Four approaches to multiplexing have been used in a wide range of communication systems. These include:

1. Frequency Division Multiplexing (FDM)

   A channel is set up by dividing the frequency spectrum into a number of parts, and dedicating a particular part to each of the users over all of the available time.

2. Time Division Multiplexing (TDM)

   A channel is set up by dividing the available time, allowing each user to use all of the frequency spectrum during its allocated time slot.

3. Space Division Multiplexing (SDM)

   A channel is set up by dedicating a physical channel in space, such as a cable or a microwave link, for example, to a particular user.

4. Code Division Multiplexing (CDM)

A channel is set up by providing different users with different signals which are distinct in form, and can be identified by a receiver even though all of the signals use all of the available time, frequency and space spectrum.

The purpose of this paper is to introduce a new type of codes, called "composition codes", and a new decoding algorithm, which can be used to effect code division multiplexing for multiple user communications. The approach is applicable to multiple access systems, broadcast systems, two user systems, or any combination of these.

The technique introduced here provides for both privacy and security, but this paper does not include an analysis of the cryptographic potential of the codes.

Section 2 describes the channel assumed in the study. Section 3 introduces the basic signaling technique, and describes the coding. The decoding algorithm, in a noise-free environment, which is the heart of the scheme, is then introduced in Section 4. In Section 5 the algorithm is extended to the case where noise must be considered. Section 6 discusses the generation of codes which are effective with the algorithm. Section 7 discusses the use of random codes as an alternative to deterministic interference-free codes. Section 8 addresses the problem of word error correction. In Section 9 we discuss some of the possible system configurations which might make use of this approach. Section 1Ø presents an example of a practical system application, specifically a local area network designed to interconnect a number of users in Space Station. Finally, Section 11 presents a number of conclusions, and suggests some areas for further study.

2. THE MULTIPLE USER COMMUNICATION CHANNEL

In this section we describe the multiple user channel configuration, and make a number of assumptions about the channel.

2.1 System Configuration

The general multiple user communication channel considered here is shown in Figure 2-1.

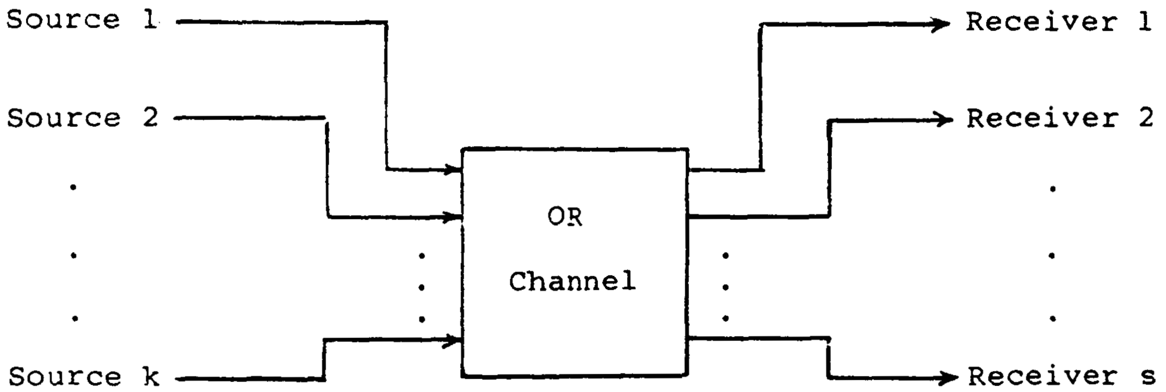


Figure 2-1. Multiple User Communication Channel

Table 2-1 defines three different basic types of multiple user configurations in terms of the number of sources k and the number of receivers s. Thus, for example, in the broadcast mode there is just one source but more than one receiver.

The codes described in this paper are applicable to any of these forms, or to any combination of them, where k and s, and hence the configuration, are either fixed over the life of the system, or are dynamic, changing with the needs of the users.

Configuration	k	s
Broadcast	1	>1
Multiple Access	>1	1
Two User	1	1

Table 2-1. Multiple User System Configurations

2.2 Assumptions

The channel assumed here is a so-called "OR" channel, in which the channel output is a zero (or a "space") if all of the inputs are zero, that is if none of the k users radiates a signal into the channel, and the channel output is a one (or a "mark") if one or more of the users radiates into the channel. In a logical sense this is equivalent to an "inclusive or" operation. The output is zero only if all inputs are zero. Otherwise it is a one.

Other channels might be considered for the coding scheme introduced here. For example, an "adder" channel tells the receiver how many users radiated into the channel. This information can be used to increase the overall rate at which data is transferred. Nonetheless, the OR channel was selected for analysis here because it is simple, and because it is robust in a noisy environment. The general coding and decoding technique could be extended to other channels if desired.

In the initial description of the coding scheme and the decoding algorithm the channel is assumed to be noise-free, so that the description can concentrate on the interference-rejection features of the system. Also, it is common to assume in many multiple user systems that the interference effects dominate noise. However, later it is shown that the algorithm can be modified to correct for random errors in a noisy environment.

3. THE SIGNALING SCHEME

This section describes the signaling which is assumed here, and defines the concepts of good and bad codebooks.

3.1 Signaling

The basic signaling technique considered in this paper is shown in Figure 3-1. During each of r time periods in a block of time of length T a given user transmits exactly one signal burst of one frequency, indexed from 1 to m. Thus each user creates a block of "on-off" or "mark-space" signals which represent the message to be sent. Each of the other users in the multiple access system simultaneously transmits some other block. The blocks which are transmitted are taken from a codebook which gives unique codewords or patterns to each user. The blocks are combined in the OR channel to form what we shall refer to here as a "composite signal". The task of the receiver is to deduce from the composite signal which blocks must have been sent by which users. It must do this without error, in the noise-free case, in spite of the fact that some users will in general radiate into the same cell in Figure 3-1 at the same time causing interference

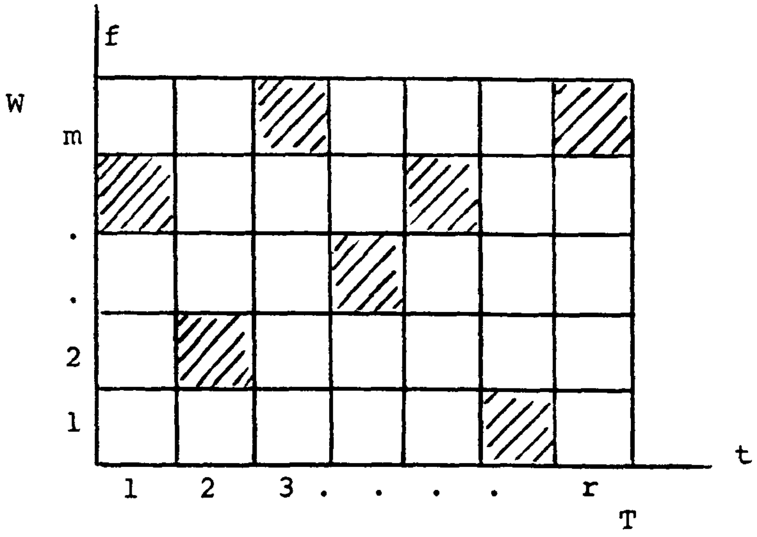


Figure 3-1. General Signaling Scheme

or a "hit". This paper shows how a decoding algorithm can accomplish this error-free decoding with reasonable efficiency. We also consider a coding scheme in which errors due to interference are permitted.

We assume first that the system has both block and bit synchronization. Block synchronization means that each of the users transmits in the same relative time frame. Everyone transmits the first set of frequencies in its codeword at the same time, from the receiver's perspective, the second set next, etc. Bit synchronization means that there is no overlap of signal duration from one signaling time to the next. Thus bits which are contiguous in time do not interfere.

We also assume that both adjacent channel interference and intersymbol interference can be neglected. The result of the above assumptions is that the only interference effect considered here is simultaneous use of the same channel. We shall refer to this phenomenon as a "hit". This is a normal and expected phenomenon, the effects of which are eliminated by the decoding algorithm.

3.2 Interference-free Codebooks

To illustrate the operation of the system, we give next a very basic example. We assume that there are just two users, called X and Y, which can send either of two messages, called A and B. (Actually, A and B need not be the same information for the two users.) The chosen code uses four frequency slots (m=4) and two time slots (r=2). Each of the users is assigned a codeword for each message. Each codeword has a certain number of terms or "elements" which are the frequencies transmitted in each of the time slots. A listing of the codewords assigned to the users is called a "codebook". The codebook for this example is given in Table 3-1.

Message	User X	User Y
A	1,2	1,3
B	3,4	2,4

Table 3-1. Codebook for Two-User System

For example, the element pair "1,3" in the table means that User Y sends Message A by transmitting Frequency 1 in time slot 1 and Frequency 3 in time slot 2. In Figure 3-2 below we show the composite signal matrices seen by the receiver for the four possible combinations of messages sent by the two users. The expression XA - YB means that User X sent Message A and User Y sent Message B.

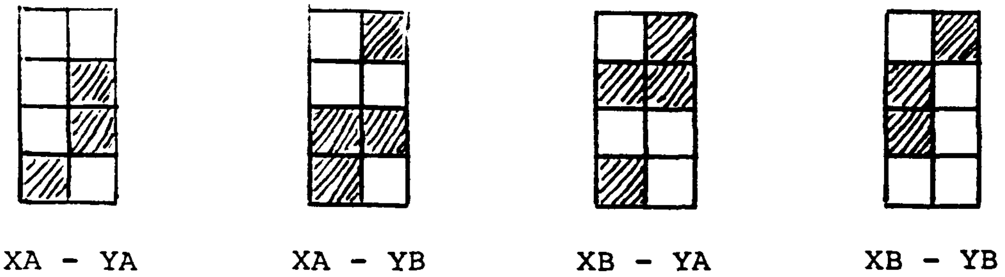


Figure 3-2. Composite Signal Matrices Seen by Receiver

At this point we introduce a definition which concerns codebooks, and which we will need later when we introduce the decoding algorithm.

Definition 1: A codebook is said to be "interference-free" if every codeword has at least one element which does not belong to any composite to which the codeword itself does not belong.

The codewords in Table 3-1 were carefully chosen so that the codebook would be interference-free.

If the codebook is changed to that shown in Table 3-2, we no longer have an interference-free code. For example, codeword YA does not have an element which does not belong to the composite XB - YB. We say then that the codeword YA has been "masked" by the interference because the composite contains all of the elements of the codeword. We shall see later that it is impossible for the basic decoding algorithm to eliminate a codeword which has been masked, and hence an error is made.

Message	User X	User Y
A	1,3	2,4
B	2,3	1,4

Table 3-2. A Codebook Which is not Interference-free

This simple example raises a basic problem which must be addressed. Suppose that we wish to add more users or messages to the system, that is, enlarge the codebook, in such a way that we are sure that the codebook is interference-free, which is a necessary condition for the successful operation of the decoding algorithm. The complexity of this problem is made evident by considering a system with k users, each of whom has j message forms. The number of composite signals which can be created and which must be tested to ensure that the codebook is interference-free is:

$$n = j^k \tag{3-1}$$

If j and k are large, n can be extremely large, and codebook generation may be quite difficult. Before we address the problem of generating interference-free codebooks in Section 6, we turn first to the development of an algorithm which can decode composite signals into their components.

4. THE COMPOSITION DECODING ALGORITHM

In this section we introduce an efficient algorithm to detect multiple access signals of the form assumed in this paper. We also briefly describe a variation of the algorithm which could be used for the ADDER channel.

4.1 The Decoding Task

At first glance the decoding task of the receiver seems formidable indeed. The receiver must consider a composite signal matrix, looking something like Figure 3-1, except with many more filled squares or marks. It must deduce what combination of messages was sent by the users. There are two brute force approaches to the task. First, the receiver can store all j^k composite signals in memory, comparing each in turn with the received composite until the correct match is found. The approach has two problems. First, it will certainly be very tedious if j^k is at all large, requiring a significant amount of memory and processing time. Second, if the assumption that noise can be neglected is not realistic, the approach probably is not very robust. We shall return to this point when we consider the proposed algorithm.

The second brute force approach is to store in the receiver each of the codewords in the codebook, create all possible composites, one at a time, and then compare each with the received composite. This approach requires much less memory, but a great deal more computation time than the first method. For any reasonably large system either of these approaches will be extremely tedious, probably making the system infeasible. We need an approach which is far more efficient.

Let us return to Figure 3-1 to consider the dilemma faced by the receiver. Suppose that it finds that a particular square in the matrix is filled. It does not know how many users or which users radiated into that particular square. It is confused by the mutual interference phenomenon. The solution is suggested by an earlier theoretical study [Healy(1982)]. There it was shown that information is transferred to the receiver only if exactly zero or exactly one users radiate into a given square. The solution to the decoding problem is to concentrate not on the filled squares, but on the empty squares. Empty squares contain a great deal of information. They tell us that *the set of messages sent does not include any message with an element in that particular time and frequency square*. When we realize this, the decoding algorithm becomes quite simple.

4.2 The Composition Decoding Algorithm

We introduce the term "composition decoding algorithm" to identify the procedure described next. The receiver successively considers each time segment in the block (the columns in Figure 3-1). To begin, it looks at the first column, noting which frequency squares are spaces. It then goes through the codebook and deletes every codeword which has an element corresponding to a space in the received composite signal, since these codewords could not have belonged to the set which made up the true composite signal. It then goes to the next time segment and repeats the process, deleting more codewords. After it has done this for all r time segments, there can be one and only one message left in each of the columns of the reduced codebook, and this message is that which the indicated user sent. The algorithm has successfully decoded the multiple access signal in spite of the mutual interference. In this paper we shall refer to a reduced codebook which has just one message for each user as a "perfect reduced codebook". It may be that the algorithm will be able to produce a perfect reduced codebook by considering less than r time columns, depending on the particular set of signals which was sent. However, the algorithm cannot fail to yield the perfect reduced codebook if it does use all r time slots. We shall prove this point shortly, but first we consider a simple example of the application of the algorithm.

Let us return to the example shown in Figure 3-2. Suppose that the composite signal observed by the receiver is the second from the left. The receiver notes that in the first time slot Frequencies 3 and 4 are blank. It then goes through the codebook matrix and deletes message XB since that message has an element in Frequency 3 in the first time slot. The reduced matrix is shown in Table 4-1.

Message	User X	User Y
A	1,2	1,3
B		2,4

Table 4-1. Reduced Codebook

The receiver next notes that Frequencies 1 and 3 are blank during time slot 2, and hence it deletes YA, yielding the perfect reduced codebook shown in Table 4-2. The receiver then declares that User X sent Message A and User Y sent Message B.

Message	User X	User Y
A	1,2	
B		2,4

Table 4-2. Perfect Reduced Codebook

As a second example, assume that the receiver observes the composite signal on the far right in Figure 3-2. It notes that Frequencies 1 and 4 are blank in time slot 1, and hence it deletes both XA and YA. The reduced codebook is perfect, and the second time slot need not be checked.

4.3 The Decoding Algorithm Theorem

Next we show that the algorithm must be successful if the code is interference-free. We assume the following situation. The active users each select one codeword and transmit it. The receiver sees a composite signal. For the particular received composite, the actual codewords which were sent by the users are called "proper codewords". This set will in general change every time block of length T as users select new messages to send. The codewords which did not make up a particular composite are called "improper codewords".

Theorem 1: For the algorithm to decode composite messages successfully, producing a perfect reduced codebook, it is a necessary and sufficient condition that the codebook be interference-free.

Sufficiency Proof:

The algorithm removes only those codewords which have elements which are spaces in the composite signal. But proper codewords create marks where their elements appear. Hence the algorithm cannot remove any proper codewords. Furthermore, every improper codeword must, by the definition of an interference-free codebook, have at least one element where the composite has a space. The algorithm will remove all of these improper codewords. Thus, an interference-free codebook is sufficient for yielding a perfect reduced codebook.

Necessity Proof:

If the codebook is not interference-free, the algorithm will still leave all of the proper codewords in the reduced codebook. However, it will also leave all of those improper codewords which do not have at least one element which does not belong to the composite in question. Hence, a perfect reduced codebook is not obtained, and the condition is seen to be necessary.

4.4 The Decoding Algorithm for an ADDER Channel

In an ADDER channel it is assumed that the receiver knows how many users radiate into a given signal element. For this channel the decoding algorithm is exactly the same as above. If the code is interference-free and there is no noise, there can be no errors. The advantage of the ADDER channel is in its enhanced ability to correct errors due to interference and noise, as we shall see in Section 8.8.

5. THE DECODING ALGORITHM IN A NOISY ENVIRONMENT

In the last section we introduced an algorithm which efficiently detects multiple access signals without ambiguity despite their inherent mutual interference. We assumed there that the noise was negligible compared with the interference. In this section we extend the algorithm to the case where noise is not neglected.

Before proceeding we should note that there are a number of ways of correcting errors made in detecting user messages or "words". In Section 8 we return to a study of approaches to word error correction. In this section we show only how the composition decoding algorithm can treat this problem.

5.1 The Effects of Noise

Consider once again the matrix of Figure 3-1, and assume that it represents the composite signal seen by the receiver. Each of the squares represents one bit of transmitted data. In this paper we have assumed a frequency hopping signaling scheme. However, the general logic of the argument can be applied to any binary signaling scheme. For any one of the squares (bits) in Figure 3-1, two types of error can be made due to the presence of noise and interference. If the bit is supposed to be a mark but the noise drives the signal below the detection threshold, a space is indicated and we have an error called a "deletion". If the bit is supposed to be a space but a mark is estimated, the error is called an "insertion". Because of these two forms of error, the composite signal matrix which is manipulated by the receiver may not correspond to the set of actual transmitted signals. Ordinarily we will expect the matrices to be quite similar, however, since the bit error probablility is usually quite low.

5.2 A Variation in the Composition Decoding Algorithm

To form the matrix it is to work on, the receiver examines the signal in each frequency band for each time slot, and compares the signal level with a predetermined threshold. It then makes a "soft" decision about the signal which was sent. That is, rather than make a "hard" decision that the signal was either a mark or a space, it measures the distance between the signal and the threshold, and stores this as a quantized number. This is a measure of the confidence that the receiver has in its decision. If a signal is very near the threshold - a borderline decision - the receiver will have much less confidence in its decision than if the signal is far from the threshold. For each square marked as a space in the working matrix, the receiver stores the location of the square in a "Deletion Candidate Buffer", with the square for which the signal was nearest the threshold taking the highest place in the ordered buffer. This square is then the prime candidate for the location of a deletion if one is suspected. Similarly, the receiver has an "Insertion Candidate Buffer" to store the location of the ordered candidates for an insertion if one is suspected.

Once the Candidate Buffers have been loaded, the receiver builds a working "hard" matrix with the squares filled or empty based on whether the signal was above or below the threshold. It then applies the algorithm described in Section 4 above. At the end of the algorithm, after the r time slots have been inspected, the receiver counts the messages which still remain in the reduced codebook. It produces a vector $\bar{D}$, called the "Decision Vector", with the elements of the vector being the number of messages associated with each of the k users in order. Thus, the vector has the form:

$$\bar{D} = (d_1, d_2, \ldots. , d_i , \ldots. , d_k) \qquad (5\text{-}1)$$

where d_i is the number of messages remaining in the row of the reduced codebook associated with user i. If the reduced codebook is perfect, as defined in Section 4, $d_i = 1$ for all i. That is, the Decision Vector takes a form which we shall call a "Unity Vector":

$$\bar{U} = (1, 1, \ldots. , 1 , \ldots. , 1) \qquad (5\text{-}2)$$

If the Decision Vector equals the Unity Vector ($\bar{D} = \bar{U}$), the receiver declares that the detection is complete and the messages are those given in the perfect reduced codebook. It is extremely unlikely that there could ever arise a situation in which $\bar{D} = \bar{U}$ and yet still have message errors. This would require highly unlikely coincidences to produce the cancelling effects to create a perfect reduced codebook from faulty data. It seems that this phenomenon can be ignored.

If $\bar{D}$ is not the unity vector, then we must have faulty primary data due to bit errors, either deletions or insertions or both. To facilitate the discussion, we define three conditions and a symbology to represent these.

$\bar{D} > \bar{U}$

$\bar{D} < \bar{U}$

$\bar{D} <> \bar{U}$

The first expression, read as "$\overline{D}$ is greater than $\overline{U}$", means that at least one of the elements of the $\overline{D}$ vector exceeds 1, and none is zero. The second expression, "$\overline{D}$ is less than $\overline{U}$", means that at least one element of $\overline{D}$ is zero and none exceed one. The final expression means that $\overline{D}$ has at least one element equal to zero, and at least one greater than one. Furthermore, we shall want to include the possibility that for a given condition it is either true that $\overline{D} = \overline{U}$ or that $\overline{D} < \overline{U}$. This is represented by $\overline{D} \leq \overline{U}$. We shall also have use for the form $\overline{D} \geq \overline{U}$, with obvious meaning.

If the matrix experiences one or more deletions due to noise, then $D \leq U$. The reason is obvious. If extra deletions appear in the working matrix, the algorithm will have more spaces than if no deletions are made, and hence when it searches for signals which could not have been sent, it will delete at least as many as in the case of no deletions.

On the other hand, if the matrix experiences one or more insertions, then $\overline{D} \geq \overline{U}$., since the algorithm cannot delete less messages than it did in the error-free case.

When we invert the problem, to give it the perspective of the receiver, things are not quite so simple. If the receiver finds that $\overline{D} \leq \overline{U}$, it probably can infer that one or more deletions have been made. If $\overline{D} \geq \overline{U}$, then it is probable that one or more insertions were made. If $\overline{D} <> \overline{U}$, which is not likely in the first place, then it means that at least one deletion and at least one insertion have occurred. The reason that we must add the word "probably" above is that it is always possible that a combination of deletions and insertions could lead to one of the two conditions $\overline{D} \leq \overline{U}$, or $\overline{D} \geq \overline{U}$, rather than $\overline{D} <> \overline{U}$. However, such a situation is very unlikely, particularly if the bit error probability is small.

5.3 Error Correction

It is clear from the above that the receiver has powerful error detection capacities. But it is also able to correct errors as we see in the following.

If the receiver finds that $\overline{D}$ is less than $\overline{U}$, it determines from the Deletion Candidate Buffer the square which is most likely to have a deletion. It puts a mark in this location and repeats the algorithm. If the result is that $\overline{D} = \overline{U}$, the receiver declares that the messages have been detected. If $\overline{D}$ is still not equal to U, the receiver replaces the first space, and places a mark in the second location indicated in the Deletion Candidate Buffer. This process continues until $\overline{D} = \overline{U}$. If only one deletion has occurred, the algorithm must eventually correctly decode the signal, assuming that we have sufficient processing time available to work through the entire Deletion Candidate Buffer. Hence, this algorithm is capable of correcting one deletion error. If more than one deletion has occurred, this algorithm as stated will not correct them, but the algorithm can easily be modified to consider probable pairs or triples of deletions, if processing time permits.

In a similar manner the receiver can detect and correct insertion errors.

A problem arises in the multiple access configuration if a user ceases transmitting. The receiver is fooled into thinking that detection was incomplete since $\bar{D}$ was less that $\bar{U}$. However, this could be handled in two ways. First, users might be required to send "preamble" and "postamble" signals which would allow the receiver to sign them on and off, and hence neglect the resulting Ø in the Detection Vector. Alternatively, the receiver could simply remember from block to block whether a user was active or not, and use that fact in its interpretation of $\bar{D}$. It might make a mistake on the first and last algorithm passes, and waste time going through the Candidate Buffers, but it would then know the correct status of the system, which it could remember.

5.4 Additional Features of the Algorithm

Because it is a spread spectrum system, this signaling and detecting approach shares with other such approaches the ability to resist frequency selective fading to some extent, since it is possible that only some of the elements of a particular pattern will be affected by a fade. The algorithm treats the lost bits as deletions.

In addition to its capacity to eliminate the effects of interference, and detect and correct errors due to noise and fading, the receiver has other signal processing strengths which might be exploited. For example, if the receiver knows both signaling and error rate statistics, it could deduce from the number of iterations of the algorithm what is the probability that the signal estimates are correct. It could then flag results which have an error probability below some set level. It could also perhaps deduce where fading was occurring in the spectrum, and take some appropriate steps. Since the receiver knows the status of the user mix, it could perform system housekeeping tasks such as allowing some users to increase their transmission rate, logging the use time of various users, billing, etc. It could also keep statistics on where deletions and insertions tend to occur, and hence optimize the system dynamically by adjusting thresholds, switching carrier frequencies, etc. None of these are necessary to the operation of the decoding, but they might be attractive side benefits of the system.

6. THE GENERATION OF INTERFERENCE-FREE CODES

If the decoding algorithm is to decode composite signals successfully, it is necessary that interference-free codes be obtained. We begin this section with a technique for systematically generating interference-free codes which are large. We then consider the relation of these codes to other codes which have been reported in the literature.

6.1 Prime Number Codes

Tables 6-1 and 6-2 give examples of two simple interference-free codes. For reasons which will be discussed shortly, we refer to these as "prime number codes" of order r, where in these examples r is 2 and 3 respectively.

Message	User Ø	User 1
A	Ø Ø	Ø 1
B	1 1	1 Ø

Table 6-1. Prime Number Codebook (r=2)

Message	User Ø	User 1	User 2
A	Ø Ø Ø	Ø 1 2	Ø 2 1
B	1 1 1	1 2 Ø	1 Ø 2
C	2 2 2	2 Ø 1	2 1 Ø

Table 6-2. Prime Number Codebook (r=3)

The reader may wish to establish by exhaustive test that these codes are in fact interference-free. This is quite easy for r=2, with its four composites, but becomes more tedious for r=3, where

27 composites must be investigated. We shall prove shortly that prime number codes are interference-free. First we define the concept, and produce a code generation algorithm.

A prime number code is defined here as a code which provides r codewords, each of which has r elements, to each of r users, where the j^{th} codeword of the n^{th} user is the vector:

$$\overline{c_{jn}} = [j,j+n,j+2n,\ldots,j+(r-1)n], \qquad j,n=Ø,1,2,\ldots,r-1 \tag{6-1}$$

where r is prime, and where the algebraic operations in Equation 6-1 are modulo-r.

The reader may wish to verify that the codebooks given in Tables 6-1 and 6-2 follow this definition. We give one additional example of a prime number code, in Table 6-3, which is intended to help explain the discussion which follows.

Message	User Ø	User 1	User 2	User 3	User 4
Ø	ØØØØØ	Ø1234	Ø2413	Ø3142	Ø4321
1	11111	1234Ø	13Ø24	142Ø3	1Ø432
2	22222	234Ø1	2413Ø	2Ø314	21Ø43
3	33333	34Ø12	3Ø241	3142Ø	321Ø4
4	44444	4Ø123	413Ø2	42Ø31	4321Ø

Table 6-3. Prime Number Code (r=5)

We consider now the general case where r is any prime number. The column of vector codewords produced for User Ø can be written as an r-term constant vector:

$$\overline{c_{jØ}} = [j,j,j,\ldots,j] \tag{6-2}$$

where j is the index for the chosen codeword.

The row of vector codewords for the r users corresponding to message Ø can be written as:

$$\overline{c_{\emptyset n}} = [\emptyset,n,2n,\ldots,(r-1)n] \tag{6-3}$$

where n identifies the n^{th} user. From Equations 6-1, 6-2, and 6-3 it is clear that the j^{th} codeword of the n^{th} user can be expressed as:

$$\overline{c_{jn}} = \overline{c_{j\emptyset}} + \overline{c_{\emptyset n}} \tag{6-4}$$

That is, any codeword is the modulo-r sum of the vector on the left end of the codeword's row and the vector on the top of the codeword's column. For example:

$$[3\emptyset 241] = [33333] + [\emptyset 2413] \tag{6-5}$$

We shall return to an alternative way of viewing prime number codes, which arises from Equation 6-4, in Section 6.4. But, first we establish some important properties of these codes which lead to a proof that prime number codes are good.

6.2 Properties of Prime Number Codes

First, we need to recall an important property of prime numbers. We wish to consider a sequence of r numbers taken from the integers Ø to r-1 in the following way. Let the first number of the sequence be any number from Ø to r-1. Let the second number be the first number plus, modulo-r, the number n where n is any integer from Ø to r-1. The third number in the sequence is the second plus n again, and so on. Hence, the sequence is:

$$\overline{X} = [a,a+n,a+2n,\ldots,a+(r-1)n], \quad a,n=\emptyset,1,2,\ldots,r-1 \tag{6-6}$$

To see this graphically, consider Figure 6-1, for the case where r equals five.

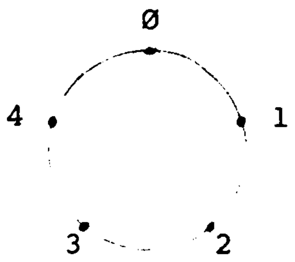


Figure 6-1. Modulo-5 Integer Circle

The sequence is formed by starting at any number and moving clockwise by n steps at a time. Hence, for example, if a=3 and n=2 in Equation 6-6 we obtain:

$$\overline{X} = [30241] \qquad (6\text{-}7)$$

The property which is of interest to us here is that as long as r is prime, all of the r elements of a sequence generated in this way will be different, unless n is Ø in which case they are all identical.

Hence, by comparing Equations 6-1 and 6-6, we see that all of the elements of any codeword in a prime number codebook are either identical, in the case of User Ø, or they are all different. (See Table 6-3.)

A second important property of prime number codebooks is that within any column (a given user) the elements of any codeword differ from those of any other codeword in all r places. This is evident from Equations 6-2 and 6-4 since addition of $\overline{c_{j0}}$ to $\overline{c_{0n}}$ has the effect of adding j to each element. The algebraic difference between any two codeword vectors with indices j_1 and j_2 is:

$$[j_2-j_1,\ j_2-j_1,\ldots,j_2-j_1]$$

Hence none of the elements is the same.

An important relation also holds between codewords of different users (different columns). Consider any two codewords in the codebook, such as:

$$\overline{X_1} = [j_1,j_1+n_1,j_1+2n_1,\ldots,j_1+(r-1)n_1] \qquad (6\text{-}8)$$

$$\overline{X_2} = [j_2,j_2+n_2,j_2+2n_2,\ldots,j_2+(r-1)n_2] \qquad (6\text{-}9)$$

In order to see if any of the elements are the same, we form the difference of the two vectors, and search for zeros.

$$\overline{X_2}-\overline{X_1} = [j_2-j_1,j_2-j_1+(n_2-n_1),j_2-j_1+2(n_2-n_1),\ldots \\ \ldots,j_2-j_1+(r-1)(n_2-n_1)] \qquad (6\text{-}10)$$

If n_2 and n_1 are equal (same user), each difference element is j_2-j_1, and hence all r elements are different as we argued above.

If n_2 and n_1 are different, then we can consider that j_2-j_1 is equivalent to a in Equation 6-6, and (n_2-n_1) is equivalent to n. Hence, the sequence obtained in Equation 6-1Ø has exactly one zero. That is, any two codewords of different users have one element in common, and differ in exactly r-1 elements. (The reader may wish to check some examples in Table 6-3.)

In summary, any two codewords of a prime number codeword differ in all r places if they belong to the same user, and in exactly r-1 places if they belong to different users. With these very important properties we are now ready to prove the following theorem.

Theorem 6-1:Prime number codes are interference-free.

Proof: Recall that a code is interference-free if every codeword contains at least one element which does not belong to any composite to which the codeword does not belong. We focus our attention on any one given codeword, called a "test codeword". Now we begin to form a typical composite by selecting a codeword from the same user as that of the test codeword. From the properties above, the test codeword has r elements which do not belong to this first member of the composite. Now we add a second codeword, from some other user, to the composite. Since this codeword has r-1 elements different from the test codeword, the test codeword must have exactly r-1 elements not in the two word composite. We add a third word from another user, and again r-1 elements are different from the test codeword. Hence, the test codeword differs from the composite in at least r-2 places since this third codeword can have no more than one element in common with the test codeword. When we add a fourth word, there must be at least r-3 elements in the test codeword not found in the composite. Continuing in this way, by the time we add the r^{th} codeword to the composite, there must be at least one element in the test codeword which is not in the composite. Since this condition defines a good code, prime number codes are interference-free.

(Note. In the above proof the expression "at least" appears in the argument when three or more codewords make up the composite because it is possible that the common element between the test codeword and a new addition to the composite was also common between the test codeword and an earlier member of the composite. When this occurs, the number of places in which the test codeword and the new composite are different does not decrease by one. Hence, when the composite is complete, a given test codeword may have more than one element not found in the composite.)

We conclude this section with a final theorem which is interesting, and which will be useful in the next section.

Theorem 6-2: Every composite of a prime number code has one and only one complete row. (Marks in all time slots at a given frequency.)

Proof: Clearly the composite must have at least one complete row, namely the repetition message of User Ø. If the composite is to have more than one complete row, the remaining r-1 users must fill all r spaces for some frequency. But we know that all r elements in any codeword are different from each other (except User Ø who has already been accounted for). Of course there is no way in which r-1 users can contribute elements which are all different to complete a row of size r.

6.3 Spectral Efficiency of Prime Number Codes

While the above procedure yields codebooks of any desired size, for r prime, it has significant disadvantages. The primary problem is that these codes do not make efficient use of the spectrum. To formalize this point, we define spectral efficiency as the ratio of the rate at which the users as a whole transmit data, to the maximum rate at which an OR channel can be used to transmit data. The latter is simply the number of squares or elements in the signaling matrix since each of these is worth one bit (it is either a mark or a space). Since there are r users with r different messages, the rate at which they send data is r times the logarithm to the base 2 of r. Hence the efficiency is:

$$e = (r\log_2 r)/rxr = (\log_2 r)/r \qquad (6\text{-}11)$$

The value of the efficiency for a number of values of n is given in Table 6-4 below.

Number of users	Efficiency
3	Ø.528
5	Ø.474
9	Ø.352
13	Ø.285

Table 6-4. Efficiency of Prime Number Codes

The reason that these codes are so inefficient is that they permit only one frequency use per time slot. This severely restricts the number of possible patterns which are permitted.

The efficiencies noted in Table 6-4 are valid if all r users are active. If only a fraction of them are active, the efficiency must be multiplied by this fraction to get an even lower efficiency. The advantage of the above codes is that they do guarantee interference-free signaling. The disadvantage is that to obtain this gaurantee we must separate messages in form, at a substantial cost in spectral space.

It is interesting to see what happens if we make the signal space available to only one user. Suppose that we keep the restriction that only one frequency is to be transmitted each time period, and that we have an r by r signaling matrix. The single user then sends one of r frequencies in the first time slot, one in the second, and so on. The number of distinct signals which the user can sent is r raised to the r power. Hence, the signaling efficiency is:

$$\begin{aligned} e &= (\log_2 r^r)/\,rxr \\ &= \log_2 r/\,r \end{aligned} \qquad (6\text{-}12)$$

which is exactly the same as that for the r user case. Note, however, that the power transmitted by this single user is only 1/r of that transmitted by all of the users together in the r user case.

6.4 Prime Number Codes as Address Codes

In this section we introduce another way to consider prime number codes, which is interesting for two reasons. First, it introduces an alternative to the composition decoding algorithm introduced in Section 4, with certain advantages and disadvantages.

Second, this alternate approach is closely related to a number of studies which have been published over the past five years or so. It is important to show how these studies relate to the present work. This issue is addressed in Section 6.5.

Let us go back to Equation 6-4, again using the codebook in Table 6-3 to illustrate a general point. We rewrite Equation 6-4 identifying its components as message and address.

$$\overline{c_{jn}} = \underset{\text{message}}{\overline{c_{j0}}} + \underset{\text{address}}{\overline{c_{on}}} \qquad (6\text{-}13)$$

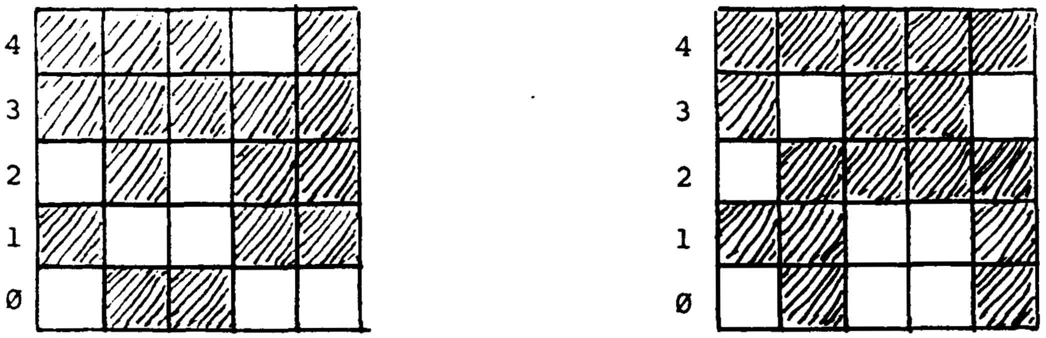
We can think of the codebook as composed not of r^2 codewords, but rather of r codewords which are found, for example for r = 5, in the left hand column in Table 6-3 and r addresses which are the sequences found in the top row, for example, in Table 6-3. Each address is assigned to one of the r users. The j^{th} codeword of the n^{th} user is then formed by adding, modulo-r, the desired user address to the desired message vector.

We shall refer here to codes which have the form of Equation 6-13 as "address codes" to distinguish them from other codes described in this paper. One feature of address codes is that they suggest a rather simple decoding algorithm.

Address Decoding Algorithm:

> To determine the codeword sent by a specific user, subtract from the index of frequencies in a given column in the composite signal the value of the address of the desired user in the given frequency column. The desired message is given by the only full row of elements in the resulting matrix.

We first give an example, and then prove that the algorithm is always successful for prime number codes. To demonstrate the process, consider a composite shown on the left in Figure 6-2 which results from the composition of five signals taken from the codebook in Table 6-3. We wish to find the message sent by User Number 3, for example, with address [Ø3142]. Since the first element of the address is zero, we decrease the index of every frequency slot in the first column by Ø, that is, there is no change. But the second address element is 3, so we decrease every frequency index by 3. This is equivalent to rolling the



Composite Signal

Decoded Composite (User 3)

Figure 6-2. Example of Address Decoding

column down by 3 steps. Since the operation is modulo-5, in this case, elements which roll down past zero appear next as 4 at the top of the next column. The decoded composite shows that the correct message is number 4.

Theorem 6-3: If the composite signal is formed from a prime number code, the address decoding algorithm yields exactly one full row, and this is the correct message row.

Proof: It is evident from Equation 6-1 that the algorithm yields the correct complete row since we obtain $c_{j\varnothing}$, the message, when we subtract the address $c_{\varnothing n}$ from c_{jn}.

To show that no other row can be full, we subtract the address for the n_1 user from the j^{th} codeword of the n_2 user, for $n_2 = n_1$.

$$\overline{c_{jn_2}} - \overline{c_{\varnothing n_1}} = [j, j+(n_2-n_1), j+2(n_2-n_1), \ldots, , j+(r-1)(n_2-n_1)] \qquad (6\text{-}14)$$

Since $n_2 = n_1$, n_2-n_1 modulo-r is the set of integers from 1 to r-1. But for any specific pair of users, n_2-n_1 plays the roll of n in Equation 6-1, and hence the codeword in Equation 6-14 is a member of the original codebook. That is, subtracting the address from a codeword yields another codeword, and hence the resulting composite matrix is itself a legitimate composite of the codebook. But, since it is a composite, it can have one and only one complete row, from Theroem 6-2.

It is appropriate at this point to make some comments on the comparison between composition codes and address codes before turning to a discussion on previous work on address codes.

In a noise-free environment the address decoding algorithm and the composition decoding algorithm are equally effective in decoding signals of individual users without error. Since the address decoding algorithm is inherently simple, it may be less complex to implement than the composition decoding algorithm in some applications. However, if we wish to simultaneously decode messages from a number of users, the composition decoding algorithm may well be less costly.

If noise is present or if the codewords are random (the codebook is not interference-free) the address decoding algorithm is less effective in correcting for errors. The reason is that the address decoding algorithm in the form given above does not make use of information about estimates of the other signals to help eliminate noise and interference effects on a given user. We shall return to this question of word error correction in Section 8.

6.5 Previous Work in Address Codes

In this section we review some of the work which has appeared in the literature, principally in the past 5 years, in the area of coded frequency hopped signals, and address codes in particular.

We begin by mentioning a recent paper by Chang and Wolf(1981) which assumes a general frequency hopping scheme much like that considered here. Two types of channel are studied, one of which is without signal intensity information, and is equivalent to the OR channel assumed here. The other uses intensity information, that is, it assumes that the receiver can tell how many signals were radiated into a particular signal cell. Techniques are given for constructing codes which are uniquely decodable in a noise-free environment. Unfortunately, the resulting codes require that all users be active if the decoding is to be correct. We do not wish to make this restriction, and hence these codes are not useful here.

The concept of address coding appears to have been introduced first in the literature by Viterbi(1978), who applied it to a proposed low-rate mobile communication system. The paper begins with an excellent discussion of the comparative advantages of frequency hopped code division multiplexing (CDMA), particularly from a spectral efficiency standpoint. Addresses are assumed to be random or pseudorandom. Error rates are obtained in noise-free as well as noisy environments. It is also shown how error rates can be reduced through coding. The paper has an important description of the effects which the modulation form and system timing have on the performance of the system.

At about the same time as the Viterbi proposal, Cooper and Nettleton(1978) also proposed a frequency hopping spread spectrum technique for mobile radio. Their work concentrates to a great extent on the modulation form. The codes assumed for the study are similar to prime number codes, with the important property that between any two codewords only one element is in common. The analysis also assumes that interference can be modeled as Gaussian noise. The proposal was critiqued by Henry(1979) who questioned the efficiency of the system, and by Mazo(1979), who questioned the assumption that interference should be modeled as Gaussian noise.

Goodman, Henry and Prabhu(1980) proposed a multilevel frequency shift keying system using coding, which is closely related to some of the approaches discussed in this paper. Specifically, they use addressing to translate repetition code messages into frequency hopped codewords such as those found in the prime number codes discussed above, and they use an address decoding algorithm to recover the signals. They obtain the probability of error for a given number of inteferring users assuming that user addresses are chosen randomly. Sometimes the interference leads to a failure to decode, yielding an error, and sometimes it does not. Hence these codes are not good in the sense defined above. But they do tend to provide for higher efficiency than interference-free codes, at the cost of non-zero error probability. We shall return to a deeper study of random codes in Section 7.

The problem of good addresses, which is analogous to the problem of interference-free codes, was studied by Einarsson(1980 and 1982). Using Einarsson's terminology, a codeword is given by the vector:

$$\overline{y_m} = \overline{a_m} + x_m \cdot \overline{1} \qquad (6\text{-}15)$$

where x_m, the address of the message to be transmitted, has the range [0,1,2,...,(Q-1)], and Q is the number of frequencies available in the frequency hopping system. The vector $\overline{1}$ is a unit vector, consisting of L 1's, where L is the number of time slots in a block. Finally, $\overline{a_m}$ is an address vector generated by:

$$\overline{a_m} = (b_m, b_m C, b_m C^2, \ldots, b_m C^{L-1}) \qquad (6\text{-}16)$$

where b_m, which is uniquely assigned to a particular user, is an element of the Galois Field of order Q, GF(Q). [For a review of Galois Fields, see Einarsson(1980), or Lin and Costello(1983).] It is necessary that C be a primitive element of GF(Q). The algebraic operations in Equation 6-16 are modulo-Q. The maximum number of codewords is Q, and the maximum value of L is Q-1. It is shown in the paper that the codewords generated by the above equation can have no more than one element in common. As with the prime number codewords, there are no common elements for a given user's codewords. Unlike prime number codes, however, the Einarsson codes have some codeword pairs from different users that have no common elements. To illustrate the use of the above generator, we repeat Example 1 from Einarsson, and expand on it, showing it's relation to the composition decoding algorithm.

Example 6-1. Let Q=7, L=4, and C=3

Applying Equation 6-16, the address for the user assigned b_m = 2 is, for example:

$$\begin{aligned}\overline{a_m} &= (2, 2x3, 2x3^2, 2x3^3)\\ &= (2,6,18,54)\\ &= (2,6,4,5)\end{aligned}$$

where the last line is the result of taking the previous line modulo-7. The codeword indexed by x_m = 4 is, for example:

$$\begin{aligned}\overline{y_m} &= (2,6,4,5) + (4,4,4,4)\\ &= (6,10,8,9)\\ &= (6,3,1,2)\end{aligned}$$

We can generate a maximum of Q=7 addresses. Each address can be applied to any of 7 repetition codewords. In this way we can create a codebook of the form of those in Section 6.1. For this example the codebook takes the form shown in Table 6-5.

We consider this example now for the case where the number of users may vary from 2 to 7. Let us start with 2 users. Recall that no pair of codewords can have more than one element in common. Hence it must be true that any codeword for a given user, except that which has been transmitted, must have at least three elements which are not found in the composite. Hence the composition decoding algorithm or the address decoding algorithm can successfully eliminate such an improper codeword. (See the proof of Theorm 6-1 above for more detail on this reasoning.) Likewise, if three or four users are active, there will still be at least two or one elements, respectively, which are not in the composite, and hence decoding is still error free. But, if five users are active, it is possible that they will contain a set of elements which is identical to that of some improper codeword. If this happens, the improper codeword cannot be removed by the composition decoding algorithm, and the address decoding algorithm will yield a second full row, producing a word error.

Message	User Ø	User 1	User 2	User 3	User 4	User 5	User 6
Ø	ØØØØ	1326	2645	3264	4513	5132	6451
1	1111	243Ø	3Ø56	43Ø5	5624	6243	Ø562
2	2222	3541	416Ø	5416	6Ø35	Ø354	16Ø3
3	3333	4652	52Ø1	652Ø	Ø146	1465	2Ø14
4	4444	5Ø63	6312	Ø631	125Ø	25Ø6	3125
5	5555	61Ø4	Ø423	1Ø42	2361	361Ø	4236
6	6666	Ø215	1534	2153	34Ø2	4Ø21	534Ø

Table 6-5. Example Codebook for an Einarsson Code

The error described above is not correctable if either of the algorithms must work with data from only one user. If, however, the signals from the others users are available, the error can probably be corrected. To see how this is done, let us return to the example above. We assume that all users are active, and that they have chosen the signal set:

$$\overline{x_m} = (2,4,\emptyset,6,1,3,2)$$

Using the codebook in Table 6-5 we obtain the composite signal shown in Figure 6-3.

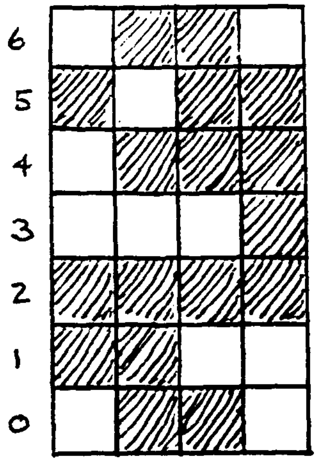


Figure 6-3. Composite Signal Example for Einarsson Code

If we now apply the decoding algorithm to this composite, we find that only one signal remains in each column except for User 3 for whom two signals are found, (1042), and (2153). If the probability of word error is low, it is very unlikely that both of these signals have been masked by the interference. It is much more likely that just the improper codeword was masked. The error correction algorithm which follows depends on this condition. If both of the words are masked, the error cannot be corrected.

The error correction concept is simple. The improper codeword is that whose elements are all found in the composite formed by the other codewords. This is why it was not removed. A quick check shows that the elements of (1042) are all found in the other codewords, including the other candidate in Column 3, but excluding (1042) of course. Hence (1042) was masked. This is not true of (2153), and hence (2153) is the correct codeword. It is very important to note that this variation in the algorithm, which is *probably* capable of correcting detected errors, requires knowledge or at least an estimate of the other signals which caused the interference. If this knowledge is available, it is also possible to alter the address decoding algorithm so that it can *probably* detect errors. We shall discuss this shortly.

Einarsson obtained a bound for the error probability for the above code.

$$P_W \leq (Q-1)(M-1)(M-2)\dots(M-L)/Q^L \qquad (6\text{-}17)$$

where M is the number of active users.

Note that this probability is 0 for any number of users up to L, as we argued above that it must be. That is, Einarsson codes are interference-free for $M \leq L$. For $M>L$ the error probability becomes non-zero and increases as the additional users provide an increasing amount of interference. Finally, the presence of Q and L in the denominator suggests that we can trade error probability against spectrum space. We shall see more about this in Section 7.

Einarsson also compared the deterministic code described above with a code in which the addresses are chosen randomly. The deterministic code exhibited slightly better performance than the random code.

It was observed above that the basic address decoding algorithm can detect but not correct errors. Timor(1981) suggested an augmented algorithm which makes use of the signals sent by other users to determine (probably) which of two or more signals indicated for a single user is correct. The augmented algorithm accomplishes essentially the same task as the variation in the composition decoding algorithm discussed earlier. Timor reported an improvement in the spectral efficiency of about 50 to 60% with the modified algorithm.

Subsequently, Timor(1982) proposed a further extension of the coding scheme in which more than one tone can be transmitted at the same time by a given user. This gives flexibility to the system designer since the time slot length can be increased. It also permits a small increase in the number of users for a given error probability.

Finally, Yue(1983) has written a review of proposals for the application of frequency hopping spread spectrum to mobile radio, covering much of the same ground as this section.

7. RANDOM CODES

An approach to the problem of the relatively poor efficiency of codes which are guaranteed to be interference-free is to relax the requirement that the code be interference-free. One way to do this is to generate random codes in which there is occasionally some interference, resulting in errors which we must either accept or correct. In this section we consider this problem for random address codes and random codeword codes.

7.1 Random Addresses

Random codes can be generated in a number of ways. One way is to maintain the signaling scheme introduced in Figure 3-1, and select address frequencies to be transmitted during each time slot randomly. This approach can be called "random address coding". It has been used in the analysis of a number of systems studied in the literature. [See, for example, Viterbi(1978), Goodman, Henry and Prabhu(1980), Haskell(1981), and Einarsson (1980).]

We consider a system which has k active users each of whom has m messages which are coded using m frequency values and r time slots. From Section 6-1 the j^{th} message of the n^{th} user can be expressed as the vector sum:

$$\overline{c_{jn}} = \overline{c_{j\emptyset}} + \overline{c_{\emptyset n}} \tag{7-1}$$

where, as before, the message is the r-term vector:

$$\overline{c_{j\emptyset}} = [j,j,\dots,j], \qquad j = \emptyset,1,2,\dots,m \tag{7-2}$$

But now the address vector is:

$$\overline{c_{\emptyset n}} = [a_1,a_2,\dots,a_i,\dots,a_r] \tag{7-3}$$

where the a_i are obtained randomly, taking, with equal probability, any value from Ø to m.

As we saw in Section 6 an error is made in address decoding if in addition to the desired full row corresponding to the message, one or more other full rows are found. We begin by considering the probability that a specific one of the m-1 non-message rows is filled by the interfering signals.

$$P_1 = [1-(1-1/m)^{k-1}]^r \qquad (7\text{-}4)$$

The logic of this equation is simple. The term (1-1/m) is the probability that a given potential interferer does not radiate into a specific time slot in the row of interest. Then the term $(1-1/m)^{k-1}$ is the probability that none of the k-1 potential interferers radiates into this frequency slot. One minus this term is the probability that at least one does radiate that frequency. This difference raised to the r^{th} power is the probability that all r time slots in the row of interest are radiated by at least one user.

In any column of m messages, an error occurs if one or more of the m-1 words other than the correct word are indicated by the address decoding algorithm. If P_1 is small, the probability that at least one of the words is in error is approximately m-1 times P_1. And in fact this is an upper (union) bound on the error probability. Hence we can write the error probability bound as:

$$P_e \leq (m-1)P_1 \qquad (7\text{-}5)$$

If we assume that there is not more than one error, we can reduce P_e to:

$$P_e \leq (m-1)P_1/2 \qquad (7\text{-}6)$$

by using a coin flip to choose between the two candidate words.

This result was originally obtained by Viterbi(1978), and was later studied in the literature cited above.

7.2 Random Codewords

In this section we consider a second appoach to random coding in which we generate each codeword randomly instead of the addresses. This is done by making a random binary decision as to whether to put a mark in each of the n=mxr elements in the signaling matrix. Hence, in any one time slot we may end up with anywhere from Ø to m frequency marks.

For convenience, let us think of taking the mxr signal elements and stringing them out in a row instead of writing them in matrix form. Then a random codebook has the form shown in Table 7-1.

	xxx...x	xxx...x	....	xxx...x
	xxx...x	xxx...x	....	xxx...x
w messages	.	.	.	.
	.	.	.	.
	xxx...x	xxx...x	....	xxx...x
	n elements			

Table 7-1. Random Codebook

Each customer is given a column of w message codewords. Each codeword, represented by xxx...x, has n elements, each of which is randomly set to be a mark with probability p and a space with probability 1-p. Thus xxx...x is a random binary sequence. It is can be seen to be equivalent to a frequency hop signal matrix if the first r binary values in the codeword make up the bottom row in the matrix (Figure 3-1), the next r values make up the second row, and so on. The actual number of elements per codeword is clearly random, and is in fact Bernoulli.

The composition decoding algorithm will be able to remove any improper codeword if it has one element which does not belong to the received composite. This is true whether the codebook is expressed in the form of binary codewords of length n=mxr, or in the equivalent m-ary codewords of length r. The probability that a particular codeword can be removed by the algorithm is the probability that the codeword has at least one element which does not belong to a composite created by some k users. For a codeword which has u elements, this is:

$$P(e|u) = [1 - (1-p)^k]^u \qquad (7\text{-}7)$$

where p is the probability that any given element in the n-bit codeword is made a mark by the random number generator. The unconditional word error probability is found by averaging over all of u.

$$P_2 = \sum_{u} P(e|u)P(u) \qquad (7\text{-}8)$$

where, once again, u is Bernoulli, with average value np.

It is interesting to consider the effect on the codebook as a whole of a codeword which happens to have a greater than average value of u. When such a word is an improper codeword, it is more likely that it will be removed as desired than will a codeword with an average number of marks. But when such a codeword is sent by its user, it causes more interference to other users than average codewords, increasing the probability that some other improper codewords will not be deleted by the algorithm. When u is less than np, the opposite effect occurs.

The above situation raises the question of strategies open to the codebook designer. One possibility would be to go through the random codewords which have been generated in search of those with large u, and give these to high priority users.

If all users have the same priority, two strategies are possible. First, we could go through the codewords which have been generated, and distribute small and large as evenly as possible among the users. Second, we could simply discard all codewords for which u does not equal np. (The latter approach has the attractive feature that it simplifies the mathematics greatly since it reduces the summation of Equation 7-8 to the simple one term form of Equation 7-7). If we use this second strategy in designing the codebook, we end up with a word error probability of:

$$P_2 = [1 - (1-p)^k]^{np} \qquad (7\text{-}9)$$

We shall further alter the random codebook by arranging to assign codewords to a given user such that all their elements are different. Then the proper codeword of the user whose improper codeword we wish to delete cannot interfere, and the error probability becomes:

$$P_2 = [1 - (1-p)^{k-1}]^{np} \qquad (7\text{-}1\emptyset)$$

If we have k users, each of which is given m messages, and we use n=mxr one-bit elements to signal, the spectral efficiency of the system as a whole is:

$$b = (k\log_2 m)/n \qquad (7\text{-}11)$$

Hence, the system efficiency goes up as k goes up, but from Equation 7-9 the word error probability also goes up. We know from theory [see, for example, Healy(1982)] that in the limit as n becomes unbounded, the channel capacity of the OR channel is a maximum when the probability that any particular element in the composite is a mark is 50%. If there are k users, this probability is:

$$1 - (1-p)^k = 1/2 \qquad (7\text{-}12)$$

Taking the natural logarithm of both sides, we have:

$$k\ln(1-p) = -\ln 2 \qquad (7\text{-}14)$$

If $p \ll 1$, $\ln(1-p) = -p$

$$kp = \ln 2 = 0.693 \qquad (7\text{-}14)$$

This observation is also made by Viterbi(1978).

We next obtain a bound on the probability that one or more errors are made in a given users column. This is the same as the form of Equation 7-6.

$$P_e' \leq (m-1)P_2 \qquad (7\text{-}15)$$

Let us compare Equations 7-4 and 7-10. If we wish the number of filled elements to be the same in random address coding and in random codeword coding, we set $p = 1/m$, and then $np = n/m = r$. Hence the two probabilities are the same. We have gained nothing by going to random codewords. At first this seems counterintuitive, since there are far more patterns which can be generated for sending messages using random codewords than with address codewords. However, when the codewords are combined to form a composite, the probability of any particular composite tends to $1/2^n$ in either case. The effect of using only one frequency per time slot is erased as we add signals to form composites.

The above result is important since it says that address codes, which are relatively simple to decode, are as good as purely random codeword codes.

7.3 Error Probability Bounds

Since we have established that random address codes and random codewords codes have identical performance for large k, we can use either one to study error bounds. We shall work with Equation 7-6. Using Equations 7-4, 7-11, and 7-14, we obtain:

$$P_e \leq (2^{bn/k} - 1)[1 - (1-1/m)^{k-1}]^{n(\ln 2)/k} \qquad (7\text{-}16)$$

If k and n=mxr are large, and we assume that Equation 7-12 applies, implying optimal filling of the signal matrix, we have:

$$P_e \leq 2^{bn/k}[1/2]^{n(\ln 2)/k} \qquad (7\text{-}17)$$

$$= 2^{(n/k)(b - \ln 2)} \qquad (7\text{-}18)$$

The exponent is negative as long as the efficiency b is less than ln2 or 0.693. This is not surprising since Cohen, Heller, and Viterbi(1971) established that ln2 is the capacity of the OR channel.

Equation 7-18 shows that the error probability goes to zero as n goes to infinity so long as the channel spectrum efficiency is less than the channel capacity.

It is important to keep in mind Equation 7-11 which determines m for a given set of b, k and n. Solving for m we have:

$$m = 2^{bn/k} \qquad (7\text{-}19)$$

Thus for a given efficiency b and block length n, the probability of error given by Equation 7-18 goes down as the number of users goes down, but the number of messages per user goes up. This increases the complexity of the decoding process, and can be a limiting factor in some situations. To quantize these effects, in Table 7-2 we give P_e and m as a function of k for a given b and n.

k	P_e	m
100	2.50×10^{-1}	32
80	1.77×10^{-1}	76
60	9.90×10^{-2}	322
50	6.25×10^{-2}	1024
40	3.12×10^{-2}	5793
25	3.91×10^{-3}	1.05×10^{6}
20	9.77×10^{-4}	3.35×10^{7}
10	9.50×10^{-7}	1.12×10^{15}
5	9.10×10^{-13}	1.27×10^{30}

n = 1,000 b = 0.50

Table 7.2 Relation of Error Probability to Number of Messages

We might hope to keep m down to a reasonable number by manipulating n and k. But in fact, for a fixed efficiency, this is not possible. This can be seen by using Equation 7-19 to eliminate n and k from Equation 7-18. Solving for m, we have:

$$m = P_e b/(b-\ln 2) \qquad (7\text{-}2\emptyset)$$

There is clearly a trade-off between m and P_e. In the next chapter we shall see that this translates into a trade-off between error correction within a single frame and error correction over a number of frames.

8. WORD ERROR CORRECTION

We have seen throughout this study that it is possible in many situations for the algorithms described here to make errors in their estimates of the words or messages sent by the various users. We have discussed some ways to correct these errors. The purpose of this section is to consider in more depth the problem of reducing the word error rate through coding, either within the basic signaling frame or over a number of frames. We begin by reviewing the causes of error, and then turn to some ways of reducing the effects of errors.

8.1 Causes of Word Errors

Errors in the decoding of words can be due to any of the following.

Interference from other users

Random noise

Fading

Jamming

If the codebook is interference-free, in the sense of Section 3, it is not possible for interference alone to cause errors using either of the algorithms discussed here. If, however, the codebook is not interference-free, either algorithm can yield an error in decoding a desired user's signal if the other users happen to send a set of signals which masks one of the improper codewords in the column. This is true whether the code is deterministic as in the case of Einarsson codes, or whether the codes are random.

The second source of error is random noise which can change a mark to a space, called a deletion, or a space to a mark, called an insertion.

Errors can also result if the signal fades selectively over the frequency bandwidth of the system. Then certain of the elements in the signaling matrix may be lost, causing deletions.

A fourth source of error is intentional interference or jamming. This problem is not considered specifically in this paper.

8.2 Forms of Word Errors

There are three basic forms of word error. These are listed in Table 8-1, along with their possible causes, and an indication of whether they can be detected by the algorithms discussed here.

Error Form	Cause	Error Detection
Word Addition	Interference or Noise	Yes
Word Subtraction	Noise	Yes
Word Addition and Subtraction	Interference and Noise	No

Table 8.1 Forms of Word Errors

Word addition means that one or more improper codewords, as well as the proper codeword, are indicated by the algorithm. This can result from interference if the other users mask one of the improper codewords, or from noise if insertions occur. Clearly the error is detected by the algorithms since more than one message is indicated.

Word subtraction means that the proper codeword does not appear in the list, and no improper codeword is indicated either. Interference cannot cause this phenomenon, since a deletion alone produces this situation. Since the receiver sees no signal, an error is detected, if of course it knows that the user is transmitting.

Word addition and subtraction means that the proper codeword is deleted, which must be due to noise, and exactly one codeword is added in the same column, which can be due to noise or to interference. This error is not detected since the receiver thinks that the single indicated codeword is correct.

8.3 Overview of Error Correction Coding

So far in this paper we have considered only single frames of data, transmitted using the signaling matrix shown in Figure 3-1, once every T seconds. It is possible to accomplish word error correction either by coding within such frames, as we have already seen, or by coding over a number of frames. We shall refer to the former as "internal coding", and the latter as "external coding". It is also possible to combine these effects. Furthermore, it is possible to transmit more than one frame at a time, and code over these. Figure 8-1 suggests a number of frames which can be transmitted simultaneously, and these sets transmitted sequentially. Coding can be over any combinations of these.

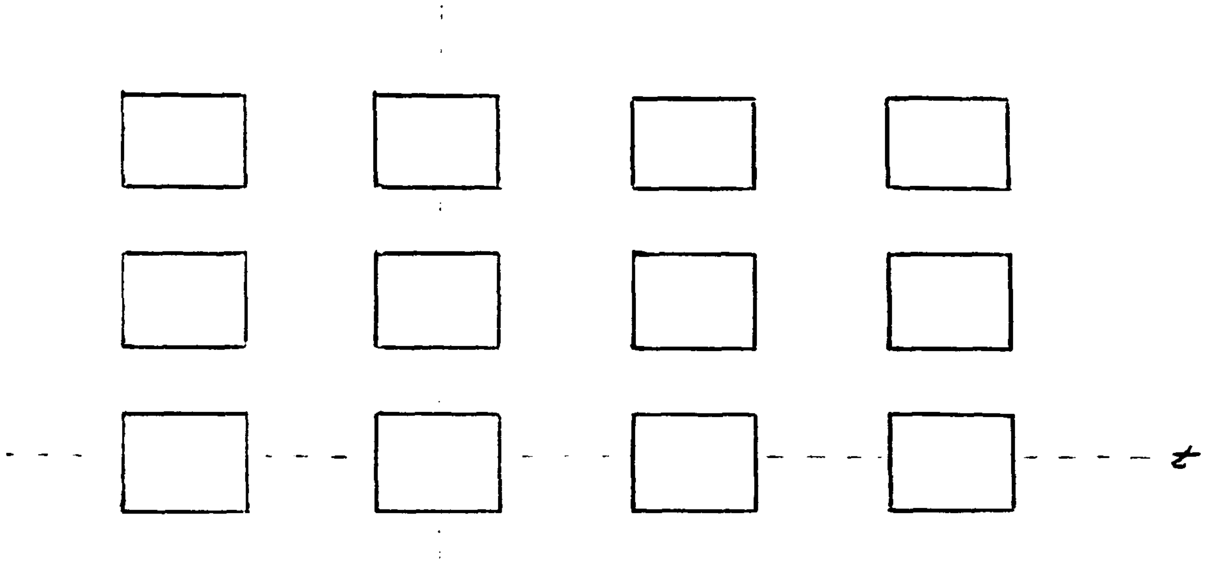


Figure 8.1 Parallel and Sequential Signaling Frames

We begin with a brief review of internal coding, which we have discussed previously, and then turn to a number of approaches to external coding.

In the development which follows we are going to assume that no more than two errors occur in a given time frame. If the probability of masking, P_1, is low, higher-order errors have little effect on the final error probabilities. The assumption greatly simplifies the analysis without significantly affecting the accuracy.

8.4 Internal Coding

An example of internal coding designed to combat the effects of noise is the use of good composition codes, along with the variation of the composition decoding algorithm introduced in Section 5. Here, a knowledge of the interfering signals and soft decision decoding are used to determine which signal bit or bits were probably in error.

A second example is the use of random codes, which cause errors due to random interference, along with the variation in the composition decoding algorithm introduced in Section 6.5. Improper codewords are removed from word addition errors by identifying which of the candidate words has elements all of which belong to the composite of the other proper codewords. This requires knowledge of all of the codewords of the other users. The composite itself cannot be used since it includes the proper codeword which of course masks itself. An alternate algorithm for removing these codewords was proposed by Timor (1981). The latter requires knowledge of the addresses of the other users, which is analogous to knowledge of the other users' proper codewords.

We saw in Section 7.1 that the probability that any one codeword is masked is:

$$P_1 = [1-(1-1/m)^{k-1}]^r \qquad (8\text{-}1)$$

Likewise, a bound on the probability of word error due to masking is:

$$P_e = (m-1)[1-(1-1/m)^{k-1}]^r \qquad (8\text{-}2)$$

Next we consider the probability that a word error is present after use of internal coding, as described in Section 6.5. Such a word error persists, if a word error occurred in the first place, with probability P_e, and if the proper codeword is masked. The latter occurs with probability P_1 since the proper codeword has the same statistical properties as any other codeword. Hence the bound on the probability that the error persists after decoding, assuming that no more than two errors can occur, is:

$$P_{eI} = P_e P_1$$

$$= (m-1)[1-(1-1/m)^{k-1}]^{2r} \qquad (8\text{-}3)$$

8.5 Parity Coding

In this section we consider a very simple parity code scheme which is effective against word addition errors due to interference, and word addition and subtraction errors due to noise. We consider a sequence of codewords transmitted by a single user. The probability of error in each of these codewords is called P_e, as in Section 7. We wish to reduce this to a smaller value P_{ep} through parity coding.

The parity code generator in the transmitter examines f consecutive m-ary codewords, and adds them modulo-m. It then transmits this sum as its f+1 message. If no word errors have been detected in the f messages, the receiver ignores the f+1 message, and declares that the indicated messages are correct. If there is no noise in the system, this declaration must be correct since interference can only cause word addition errors. If there is noise in the system, and if the parity is correct, it is virtually certain that the declaration is correct since an error could then occur only if two addition/subtraction errors had occurred which added to the same sum as the proper codewords. This would be extremely unlikely. The probability of this set of occurrences is extremely small.

If one word error occurs, and there is no noise present, the parity coding necessarily corrects the error by choosing the codeword which correctly satisfies parity. This fails only if the parity is also in error.

If two word errors appear, the receiver corrects the errors with the parity word unless the two improper codewords have the same sum, modulo-m, as the proper codewords. The probability of this is 1/m. When this situation occurs, the receiver is unable to identify the correct pair, although it is able to reduce the uncertainty to a pairwise uncertainty.

Once again, the parity is taken over f words. If g is the number of words which are in error, the probability of error after decoding is:

$$P(e|g) = (g-1)/m \qquad (8\text{-}4)$$

The probability of g errors is Bernoulli.

$$P(g) = \frac{f!}{g!(f-g)!} P_e{}^g(1-P_e)^{f-g} \qquad (8\text{-}5)$$

Hence the unconditional error probability is:

$$P_{ep} = \sum_{g} \frac{g-1}{m} \frac{f!}{g!(f-g)!} P_e{}^{g}(1-P_e)^{f-g} \qquad (8\text{-}6)$$

To reduce P_{ep} it is desirable to keep f as low as possible. However, the coding has the effect of reducing the spectral efficiency by the factor f/(f+1), which suggests that we keep f as large as possible.

As an example, we have evaluated Equation 8-3 for the case where P_e is Ø.1, and f varies from 2 to 6. In the second column P_e is shown as a general function of m. The third column is for the special case where m is 32. The fourth column gives the factor by which the spectral efficiency is reduced.

f	P_{ep}	P_{ep} (m=32)	b_f
2	Ø.Ø1Ø/m	Ø.ØØØ3	Ø.67
3	Ø.Ø3Ø/m	Ø.ØØ1Ø	Ø.75
4	Ø.Ø56/m	Ø.ØØ17	Ø.8Ø
5	Ø.Ø9Ø/m	Ø.ØØ28	Ø.83
6	Ø.135/m	Ø.ØØ42	Ø.86

Table 8-2. Error Probability After Detection (P_e= Ø.1Ø)

Thus, for example, for f equal to 5, the error probability is reduced by a factor of approximatedly 1/m, or Ø.Ø28 for m equal to 32, at a cost of reducing the spectral efficiency by a factor of Ø.83.

Returning now to the general expression for the error probability, we note that if P_e<<1, we can write:

$$P_{ep} = f(f-1)P_e{}^2/2m \qquad (8\text{-}4)$$

Using Equation 8-2 we obtain:

$$P_{ep} = \frac{f(f-1)}{2m}(m-1)^2[1-(1-1/m)^{k-1}]^{2r} \tag{8-5}$$

$$= \frac{f(f-1)}{2}(m-1)[1-(1-1/m)^{k-1}]^{2r} \tag{8-6}$$

Note that the error probability using parity coding is approximately [f(f-1)]/2 times that using internal coding.

8.6 Combining Parity Coding and Internal Coding

It is possible to combine internal coding and parity coding. The sources of error are different. Internal coding errors for some particular user arise from the set of signals chosen by the other users. Parity coding errors are a result of the particular set of f codewords chosen by the user of interest. If both coding techniques are used, parity coding is only used to correct errors if internal coding fails. Hence, if we are to have an error persist after combining the two, it is necessary that we have two persisting internal errors in f frames. This probability is given by Equation 8-4 where P_e is given by P_{eI} in Equation 8-3. Hence, the word error probability for the combination of the two techniques is:

$$P_{ec} = f(f-1)P_{eI}{}^2/2m \tag{8-7}$$

$$= \frac{f(f-1)}{2m}(m-1)^2[1-(1-1/m)^{k-1}]^{4r} \tag{8-8}$$

$$= \frac{f(f-1)}{2}(m-1)[1-(1-1/m)^{k-1}]^{4r} \tag{8-9}$$

In summary, the results given in Equations 8-1 through 8-9 are next simplified, and presented in a compact form, and the relevant terms are redefined.

P_1 - probability that any given codeword is masked due to interference from other users

P_e - probability of a word error due to masking of one of the k-1 improper codewords in a user's column

P_{eI} - probability that a word error persists after internal decoding

P_{ep} - probability that a word error persists after parity decoding

P_{ec} - probability that a word error persists after a combination of internal decoding and parity decoding

$$P_1 = [1-(1-1/m)^{k-1}]^r \tag{6-10}$$

$$P_e = (1-m)P_1 \tag{6-11}$$

$$P_{eI} = (m-1)P_1{}^2 \tag{6-12}$$

$$P_{ep} = \frac{f(f-1)}{2}(m-1)P_1{}^2 \tag{6-13}$$

$$P_{ec} = \frac{f(f-1)}{2}(m-1)P_1{}^4 \tag{6-14}$$

8.7 Error Avoidance Techniques

One of the possible applications of the coding described in this paper is to systems which act in the broadcast mode, in which a single transmitter sends a number of messages to different users. There is still in this case the possibility of interference in the form of the composite masking one or more improper codewords. However, in this case the transmitter has knowledge of and control over the composite signal. It can refuse to send composites in which it knows that one or more signals are masked. This can be done in at least three ways.

First, the transmitter could simply delete interfering or interference signals from the composite, perhaps those intended for a signal with a low priority. The cost here is a decrease in the data rate for the low priority user.

Second, the transmitter might be free to choose from more than one codeword to send a given message word, choosing those combinations which do not lead to interference. The cost here is a larger codebook at the transmitter and the receiver, increasing coding and decoding implementation costs.

Finally, the broadcaster could send an extra interference-resolving signal to all users. For example, this might take the form of a parity signal as was discussed earlier.

8.8 Error Correction with ADDER Channels

In the case of the ADDER channel it is possible to use a variation of the basic composition decoding algorithm which yields a lower error probability than can be obtained with the OR channel. Assume that the receiver knows all of the user signals or proper codewords. If the receiver detects a word error, it goes back to the decoded messages and removes all of the known codewords from the received composite by reducing the count for each signal element by one for each element of a known codeword. When it has completed this, the only remaining codeword is the desired proper codeword. Hence, this algorithm can always remove a single error whereas the algorithm for the OR channel could probably remove it. If more than one error is detected, the procedure is to remove as many codewords as are known, and to attempt to correct the errors on the basis of the signal elements remaining in the reduced composite.

If the user does not know the messages of the other users, it can still reduce the probability by eliminating that codeword which has the lower element count in the signal matrix. It is more likely that such a word is the improper codeword than that it is proper.

9. SYSTEM CONFIGURATIONS AND APPLICATIONS

In Section 3 we sketched briefly some general ways in which multiple user systems can be configured. In this section we address this question in much more detail for the particular types of coding of interest in this paper. The versatility of the approach suggests a wide variety of applications. We begin with a discussion of the transmitter modulation options which are available. We then consider the various forms of multiple user communication, particularly with respect to the types of codebooks required for different configurations. Also discussed in this section are techniques for prioritizing signals, and prospects for message security or secrecy. We conclude with a few examples of specific applications which serve to demonstrate the versatility of the approach.

9.1 Signal Modulation Forms

Much of the discussion in this paper has been in terms of an m-ary signal which hops from one frequency to another over r time slots. While this provides a convenient basis for discussing the coding and decoding, it is not at all necessary that the signals actually radiated by a transmitter be m-ary. Many alternative modulation forms will provide the same information. For example, we could send signals with m-ary amplitudes instead of frequencies, or we could send narrow pulses in any of m sub time slots within each of the r time slots.

We do not even have to use m-ary signaling. The information in the signaling scheme shown in Figure 3-1 could be represented in binary form, for example, by sending each row one bit at a time, perhaps by binary phase shift keying. Or we might send two bits at a time using quadrature phase shift keying. The possibilities are almost limitless.

9.2 Basic System Configurations

Section 3 introduced the basic types of multiple user system configurations. In this section we review these and discuss the codebook needs in each case.

Broadcast

In the broadcast mode a single transmitter wishes to send messages to a number of users or sets of users. The transmitter decides which receivers are to obtain which messages by selecting an address or a codeword column which corresponds to the desired receivers. This is analogous to putting an address on a message, indicating to whom it is to be delivered. The receiver must of course have the appropriate address or codeword column, which it applies in the decoding algorithm to the received composite.

The receiver may or may not have information about the addresses or codewords of other users. If it does, it can use this to decrease the error probability, though this will require a more costly implementation than simple decoding.

In the broadcast mode it is possible to consider some form of interference avoidance, as discussed in Section 8.6. Since the transmitter knows the composite to be sent, it knows which of its signals are masked. It may be able to avoid sending a masked signal by choosing between alternative signals with the same meaning. In this way error correction (avoidance) takes place at the transmitter.

Multiple Access

In the multiple access mode more than one user sends messages to a single receiver. Each user is given a unique address or codeword column. The receiver knows who is sending what by reason of this addressing, either explicit or implicit. If the receiver needs to detect a large number of signals at once, it may be more effective to implement the composition decoding algorithm than a large set of address decoding algorithms. If the receiver uses a complete codebook, this could be a fixed codebook with a column or address for every potential user, or it might be a dynamic codebook with a column or address called up from memory when a customer becomes an active user. Depending on which algorithm is most efficient in a given application, the receiver might work directly with a set of addresses to implement address decoding, or it might use the addresses to create columns of codewords which could be used with the composition decoding algorithm.

Two-User

In this mode one user wishes to communicate with one other over a channel which is also in use by others. This could be thought of as a degenerate case of the broadcast mode for the case where there is only one receiver. Again, as before, the receiver may have only his own address or column available, or he may have a full codebook set available to help decrease the word error probability.

9.3 Prioritization

There are a number of ways of providing priorities using the approaches discussed here. One of these was described in Section 8-6, for braodcast channels. If the transmitter sees that a low priority signal will be masked, or will mask some other signal, it does not transmit that signal.

In the case of random codeword codes, one user with a high priority is given codewords which have more marks than those of low priority users. Such words are less likely to be masked than those with fewer elements, and hence the probability of error is lower for these higher probability users.

A way to give a user with a high priority a greater data rate is to give the user the right to send more than one codeword, using different addresses or columns.

Still another approach to prioritization in the multiple access configuration is a protocol which might be called "traffic sense multiple access" (TSMA). In this mode each potential transmitter is required to observe the amount of traffic on the channel, and transmit only if it is below a level authorized by the user's priority. A convenient measure of the level of traffic is the percentage of marks in the composite signal.

9.4 Privacy and Secrecy

In multiple user communication systems, privacy is often required, such as in telephone and telegraph systems. In other cases, such as in bank transactions and business negotiations, secrecy is necessary.

The system described here has good inherent security characteristics. As long as a particular user is not given the codebook column or address of other users, it should not be able to determine the message sent to or by those users. This logic breaks down if there is only one or very few active users. If a single user were active, an eavesdropper could look at the messages sent, and either construct a codebook or deduce an address vector. These could then be used later when more users came on line. One way to avoid this would be for a system manager to transmit random elements to confuse the eavesdropper.

It might be thought that an eavesdropper, knowing that a particular form of address was in use, could go through all possible addresses, exhaustively, in search of the one that yielded a full row. And in fact this would be a problem if the codebook was good. But, if random coding were used, this strategy should not be effective in any fairly well loaded system since we would expect that filled rows due to interference would be rather common. An eavesdropper should not be able to determine whether an error due to interference had occurred, or whether the correct address had been discovered.

9.5 Examples of Specific Applications

The coding scheme described in this paper, using either the composition decoding algorithm or the address decoding algorithm, can be used in a very wide variety of applications. In this section we list a few typical applications, intending not to exhaust the possibilities, but rather to illustrate the versatility of the system.

Local Area Network

A local area network is set up to serve 5Ø nodes. The network can operate in any of three modes: multiple access, broadcast, or point to point. When a user wishes to access the system in the multiple access mode, it uses an address which identifies itself to the interested receiver. In the broadcast mode, it uses an address which is monitored by all receivers. In addition it also monitors his own unique address, and is capable of monitoring other addresses which are established dynamically for particular system configurations.

Telephone Switching

A telephone system is set up which provides an alternate to the conventional telephone system approaches of time, space and frequency division multiplexing. Each set in the system is given a unique address, called it's "telephone number". When a telephone is picked up, it automatically sends its address to its local office which is thereby notified that a call is to be placed. The local office sends back a dial tone. The sender then dials the address of the set which it wishes to call, possibly with additional long distance numerals attached. The local office loads both of the addresses into its dynamic codebook, and also establishes a link between them. It then selects a trunk, depending on the called address, and transmits calling information which eventually reaches the called set. If someone answers the phone, a connection is set up to the callee's local office, and the connection is complete. The switching is essentially accomplished through the process of setting up temporary address pairs in the transmitter/receiver nodes through which the telephone call is routed.

Satellite Relay

A satellite operates with a single channel over which CDMA signals are transmitted. Ground stations transmit at will. The satellite receiver acts in a multiple access mode. The satellite determines to whom the message is to be sent, and sets up either a point to point connection or a conference call as required. For security reasons the senders are not given the addresses of the stations called. This is determined by the satellite after the sender has identified the station to be called.

Internal Computer Communications

A computer is designed with single wires or leads instead of buses for internal communication. When one part of the unit, such as a central processing unit for example, needs to send a signal to another unit, such as a memory for example, it uses the address of the unit, and transmits its signal over the common signal lead.

10. A PRACTICAL SYSTEM APPLICATION

The purpose of this section is to study a specific system which might use the CDMA approaches described in this paper. The system chosen for study is a computer communications network to operate in a future space vehicle. A typical example would be the Space Station now under consideration for the late 1980's. The application is one which will require multiple user communications, and which has sufficient breadth to illustrate a number of the features of CDMA. We begin with a description of the system, and then turn to its analysis.

10.1 System Description

A block diagram of the system showing the basic set of users to be served is given in Figure 10-1. The system serves 50 nodes which can provide for human interface, or for interface with other information sources or sinks within the spacecraft. There are also two gateways which provide for radio wave communications with systems outside the spacecraft. The system also has an automatic system manager, which has a number of roles described below, and a file server which acts as a library of information or programs useful to the space station.

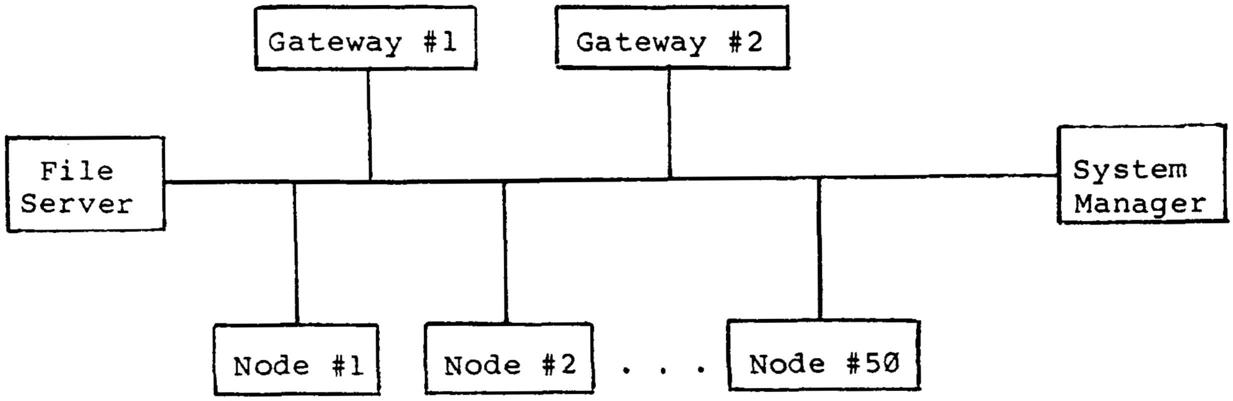


Figure 10-1. System Block Diagram

The 5Ø nodes represent the 5Ø potential users of the system. Different users are assumed to have different duty cycles, and are given different priorities, depending on the importance of their activity in the spacecraft. Related to a user's priority is it's traffic sense multiple access (TSMA) measure. This is the percent of the total number of elements in the signal matrix which are filled (marks). Users continually monitor the signal matrix. If the TSMA measure exceeds their threshold, they are not allowed to transmit. Node characteristics are summarized in Table 1Ø-1.

Node Numbers	Duty Cycle	Priority	TSMA Threshold
1-3	1ØØ%	1	1ØØ%
4-5	5	1	1ØØ%
6-15	5	2	5Ø%
16-5Ø	1	3	3Ø%

Table 1Ø-1. User Node Prioritization

Nodes 1-3 operate constantly (duty cycle = 1ØØ%). They carry vital information about the system status and safety. They are given the highest priority, and there is no TSMA threshold above which they are denied access. Nodes 4-5 operate intermittently, but they are assigned to key control personnel who also have the highest priority. They are used about 5% of the time. The next ten nodes rate a lower priority. They cannot access the system if more than 5Ø% of the signal elements were marks during the previous signal time frame. Nodes 16 to 5Ø use the system whenever less than 3Ø% of the signal elements are marks.

A typical node block diagram is shown in Figure 1Ø-2. The source message can originate from one of three types of peripheral devices: teletype terminal, physical sensor, or data storage device. The block named "source" includes whatever signal processing may be necessary to present the signal to the coding unit in digital form. For example, in the case of the sensor this means analog to digital conversion.

The coding unit takes the digital signal from the source and forms an m-ary codeword consisting of r elements. In this particular case m is 16 and r is 5. The number of codewords required to represent the output from the source depends on the

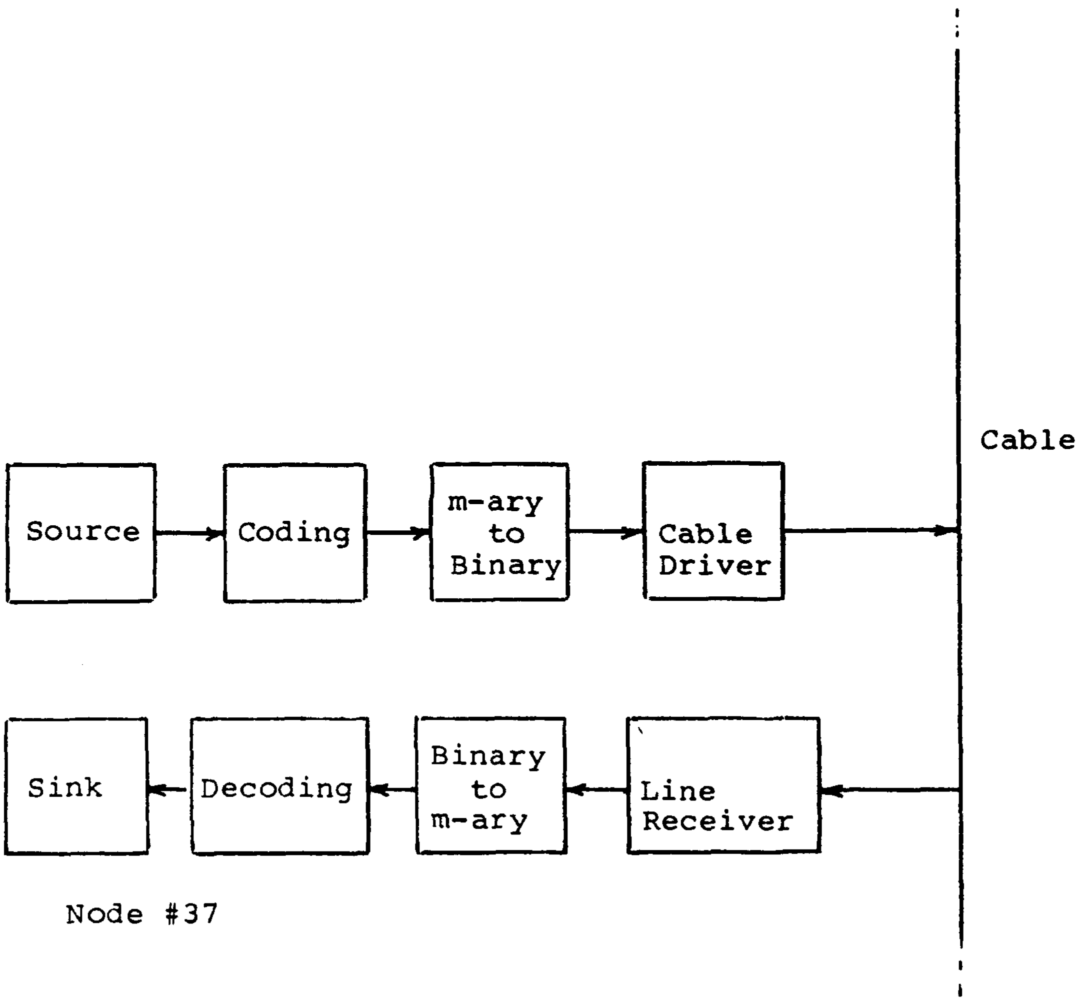


Figure 1Ø-2. Block Diagram of Typical Node

type of peripheral device in use. For example, the sampled and quantized output of the sensors is an eight bit signal, requiring two codewords. The coding unit creates the coded signal by adding, modulo-17, a five-element address to one of the following message signals determined by the source.

```
 Ø  Ø  Ø  Ø  Ø
 1  1  1  1  1
 2  2  2  2  2

   .  .  .  .

16 16 16 16 16
```

The first sixteen codewords are used to transmit four bits of data. The last codeword is used for control purposes only, as explained in more detail below.

Each of the 57 units in the system is assigned a unique address. For example, the address for Terminal#37 is:

11 4 9 13 2

In addition there is a "broadcast" address which is:

15 5 12 2 11

There are also a number of other addresses which can be assigned dynamically, as we shall see later.

The purpose of the m-ary to binary unit is to translate the 17-ary 5-element codeword into a 85 (5x17) bit binary codeword suitable for transmission on a fiber optic transmission line. The cable driver has two functions. It counts the number of marks in the 85 bit binary codeword. If the percentage exceeds the prioritized TSMA threshold, the cable driver inhibits transmission. If the count is less than the threshold, the cable driver transmits a signal into the cable during each of the appropriate 85 time slots during each frame.

The line receiver detects the composite signal consisting of all of the signals sent by the active users, including the signal which has just been put there by the line driver. The binary to m-ary unit converts the 85 bit binary signal back into a 17-ary 5-element signal which goes to the decoder.

The decoder depends on the user. For example, Terminal#37 reads only those signals directed to it, and therefore it uses a simple address decoder. However, the system manager keeps track of all messages on the line. It must be able to decode signals from all 57 devices, plus some additional dynamic sub-networks which can be set up. For this reason it uses a composition decoding algorithm, which is found to be simpler to implement than a large set of address decoders, as well as having better error correction capacities.

1Ø.2 System Protocols

This section describes the handshaking procedure the system uses to set up communication channels, and how the system operates after channels are established.

To begin a communication to one or more of the other users, a user activates a message initiation signal. For example, in the case of the teletype terminal this is done by depressing a special key called "initiate". The user then chooses one of three transmission modes: broadcast, single receiver, or conference by pressing b, s, or c, respectively. If the user presses b, a message is sent to the source to send the message (16 16 16 16 16), which is the control message, and a message is sent to the encoder to use the broadcast address (15 5 12 2 11). The encoder adds the control message to the broadcast address, and sends it to the transmission line. The user then sends two consecutive four-bit message words which identify himself as the originator of the message. These are also sent with the broadcast address. All of the 54 users monitor the broadcast channel by continually applying an address decoder to the composite signal. When they observe the control message in the broadcast channel, they know that some user is broadcasting a message, and the user's identity is given by the following two message words. With this established, the user now proceeds to send the intended message to all of the other users. At the end of the message, a sign-off message notifies the receivers of the end of the transmission.

If the sender wishes to transmit to only one of the other users, he uses the address which is unique to the intended recipient of the message. And of course all terminals continuously monitor their own address.

As a third alternative the user may request a conference call involving any desired subset of users. This is accomplished by sending the control word addressed to the system manager, who then makes any necessary arrangements for the call, including the determination of priorities, checking to see if the other users are free, and assigning a temporary address to be used during the call.

1Ø.3 Error Correction

The system has available four levels of error control which can be called up according to the interests of the users and the nature of the message. These are:

No Error Control

In this case no error control techniques are used. The decoder detects errors but cannot correct them. The system is designed to have a word error probability, without correction, of $1Ø^{-3}$. This is considered to be adequate for voice transmission and for some text transmission. Twenty of the units have no error control.

Internal Decoding Error Correction

This form of error correction is an extension of the composition decoding algorithm. It can only be used by units which decode all of the messages in the composite. In this system, this includes the system manager, and the Space Station Chief Officer's terminal, which is Terminal#4. This reduces the error probability from the uncoded $1Ø^{-3}$ to

Parity Error Correction

In this case, when a user is sending a message, every tenth word is the modulo-16 sum of the previous nine words. Terminals use this parity check to correct exactly one word error, with certainty. This level of control is employed by all of the units except the 2Ø which do not use error control. It is used by the system manager and Terminal#4 as back-up. This reduces the error probability to

Internal Coding Plus Parity Coding

If the internal decoding fails to correct the word error, parity coding is used to attempt to resolve the persisting error. This combination coding is used only by the system manager. However, other units may interrogate the system manager in an attempt to correct an error which they have not resolved.

Echo Error Correction

When a given terminal sends a message word, it simultaneously learns what interference was generated by other users by monitoring the composite signal through its receiver branch. If it sees that both its proper message and one of its improper codewords have been masked by the interference, then it knows that the composition decoding algorithm will not be able to resolve the error. So it retransmits the message codeword. This has the effect of reducing the error probability to $1Ø^{-?}$.

1Ø.4 Error Probability Calculations

The probability that an error is made due to interference depends on the number of users, which is a random variable. The use statistics for the various users isgiven in Table 1Ø-1. We assume that the users act independently, and that the probability that they will use any particular time frame is equal to their duty cycle. From Table 10-1, we have three users who use a given time frame with probability 100%, twelve with probability 5%, and 35 with probability 1%. The total number of users during any time frame is:

$$k = 3 + k_1 + k_2 \tag{10-1}$$

where k_1 and k_2 are Bernoulli, with distributions $b(k_1$: 12, 0.05) and $b(k_2$: 35, 0.01), respectively. These distributions, p(k), along with the corresponding cumulative distribution functions, P(k), are given in Table 10-2.

k	$p(k_1)$	$P(k_1)$	$p(k_2)$	$P(k_2)$
0	0.7034	0.7034	0.5403	0.5403
1	0.2487	0.9521	0.3413	0.8816
2	0.0427	0.9948	0.0988	0.9804
3	0.0047	0.9996	0.0173	0.9978
4	0.00037	0.99997	0.0020	0.9998
5	0.00002	0.99999	0.00017	0.99999

Table 10-2. Active User Probability Distributions

The distribution for k in Equation 10-1 is found by convolving the distributions of 3, k_1, and k_2. The resulting distribution is given in Table 10-3.

k	p(k)	P(k)
3	0.3800	0.3800
4	0.3478	0.7548
5	0.1775	0.9323
6	0.0539	0.9862
7	0.0117	0.9979
8	0.0019	0.9998

Table 10-3. Distributions of Total Active Users

The average number of users is:

$$E(k) = E(3) + E(k_1) + E(k_2) \tag{10-2}$$

$$= 3.95$$

Next we turn to a calculation of the word error probability. We use Equation 7-5 here, rather than Equation 7-6, since we are using error correcting techniques, rather than a coin flip, to resolve detected errors. Hence, the bound on the error is:

$$P_e = (m-1)[1-(1-1/m)^{k-1}]^r \tag{10-3}$$

$$= 16[1-(16/17)^{k-1}]^6 \tag{10-4}$$

This error probability bound is given in Table 10-4 for the significant values of k, given in Table 10-3. We also give in Table 10-4 the function P(k), which is the probability that the word error probability is equal to or less than the given P_e.

k	P_e	P(k)
3	0.0000472	0.3800
4	0.000446	0.7538
5	0.00208	0.9323
6	0.0066	0.9862
7	0.0164	0.9979
8	0.0346	0.9998

Table 10-5. Uncorrected Word Error Probabilities

The average value of P_e for this example is 1.15×10^{-3}.

So far we have only considered the word error probalility without error correction. With correction techniques applied, the error probabilities are given by Equations 8-10 through 8-14, which for this example have the form:

$$P_1 = [1-(16/17)^{k-1}]^6$$

$$P_e = 16P_1$$

$$P_{eI} = 16P_1^2$$

$$P_{ep} = 720P_1^2$$

$$P_{ec} = 720P_1^4$$

Numerical values are given in Table 10-6.

k	P_1	P_e	P_{eI}	P_{ep}	P_{ec}
3	2.21E-6	3.54E-5	7.86E-11	3.54E-9	1.74E-20
4	2.11E-5	3.38E-4	7.15E-9	3.22E-7	1.44E-16
5	9.97E-5	1.59E-3	1.59E-7	7.16E-6	7.11E-14
6	3.20E-4	5.10E-3	1.64E-6	7.40E-5	7.52E-12
7	8.04E-4	1.29E-2	1.00E-5	4.65E-4	3.01E-10
8	1.70E-3	2.74E-2	4.70E-5	2.11E-3	6.16E-9

Table 10-6. Word Error Probabilities with Correction

10.5 System Manager

The system manager is a hardware unit which has a series of tasks. In general it monitors operation of the system, and reports any major problems or defects to the chief officer of the Space Station. It also assigns any temporary address vectors which may be needed to set up ad-hoc combinations of users for conference calls, or for other reasons.

The system manager has the most powerful error correction techniques in the system. If a unit is unable to correct an essential codeword which is in error, it can interrogate the system manager. Hence, this becomes still another form of error correction available to all users.

1Ø.6 External Communications

Any user can set up a radio communications with nodes external to the Space Station by requesting a special address for that purpose, as well as access to a gateway. The address and signal forms are exactly the same as for internal communications. It is only necessary to apply signals to the relevant external radio modulators.

11. CONCLUSION

This paper has reported on three major results of a study of multiple user communication systems. These results are:

1. A new algorithm has been introduced which separates user signals in a multiple user environment characterized by mutual interference.
2. Multiple user systems have been reviewed and developed in general, and in particular it has been shown how the new algorithm relates to previous work in this field.
3. It has been shown how the approach presented here could be used to accomplish multiple user communication in Space Station.

Many questions have yet to be answered in each of these areas. Some of the most important issues which should be addressed follow.

Search for additional interference-free codes which may be more efficient than prime number codes.

Establish a stronger mathematical basis for the coding, the decoding algorithms, and the proofs of theorems.

Determine whether noise and interference are best dealt with by the same or different algorithms.

Determine the security and secrecy potential for these composition codes.

Identify additional applications of composition codes.

Study a number of external codes, and try to identify the best choice.

Study the implementation of the basic decoding algorithm and its extensions, and compare the complexity and cost with that of other approaches.

Compare the local area netword protocals described here with other protocols.

Obtain more data on the anticipated communications needs of Space Station, and improve the proposed multiple user system accordingly.

BIBLIOGRAPHY

L.A. Berry, "Spectrum Metrics and Spectrum Efficiency: Proposed Definitions," IEEE Trans. on Elect. Comp., Vol. EMC-19, No. 3, pp.254-260, August, 1977

S. Chang and J.K. Wolf, "On the T-User M-Frequency Noiseless Multiple-Access Channel with and Without Intensity Information," IEEE Trans. on Inf. Th., Vol. IT-27, No. 1, pp. 41-48, January, 1981

A.R. Cohen, J.A. Heller and A.J. Viterbi, "A New Coding Technique for Asynchronous Multiple Access Communication," IEEE Trans. on Comm. Tech., Vol. COM-19, No. 5, pp. 849-855, October, 1971

G.R. Cooper and R.W. Nettleton, "A Spread-Spectrum Technique for High-Capacity Mobile Communications," IEEE Trans. on Veh. Tech., Vol. VT-27, No. 4, pp. 264-275, November, 1978

G. Einarsson, "Address Assignment for a Time-Frequency-Coded, Spread-Spectrum System," BSTJ, Vol. 59, No. 7, pp. 1241-1255, Sept., 1980

G. Einarsson, "Coding for Multiple-Access Multilevel FSK Systems,", Tech. Rpt., TR-171, Electrical Engineering Dept., Univ. of Lund, Sweden, Sept., 1982

D.J. Goodman, P.S. Henry, and V.K. Prabhu, "Frequency-Hopped Multilevel FSK for Mobile Radio," BSTJ, Vol. 59, No. 7, pp. 1257-1275, Sept., 1980

B.G. Haskell, "Computer Simulation Results on Frequency Hopped MFSK Mobile Radio-Noiseless Case", IEEE Trans. on Comm., Vol. Com-29, No. 2, pp.125-132, Feb., 1981

T.J. Healy, "Error Correcting Codes for Frequency Hopping Multiple-Access Spread Spectrum Communication Systems", Proc. IEEE Global Telecommunications Conference (Globecom), Miami, FL, Nov., 1982

P.S. Henry, "Spectrum Efficiency of a Frequency-Hopped-DPSK Spread-Spectrum Mobile Radio System", IEEE Trans. on Veh. Tech., Vol. VT-28, No. 4, pp. 327-332, Nov., 1979

S. Lin and D.J. Costello, Jr., Error Control Coding: Fundamentals and Applications, Englewood Cliffs, NJ, Prentice-Hall, 1983

J.E. Mazo, "Some Theoretical Observations on Spread-Spectrum Communnications", BSTJ, Vo. 58, No. 9, pp. 2013-2023, Nov., 1979

U. Timor, "Multistage Decoding of Frequency-Hopped FSK Systems", BSTJ, Vol. 60, No. 4, pp. 471-483, April, 1981

U. Timor, "Multitone Frequency-Hopped MFSK System for Mobile Radio", BSTJ, Vol. 61, No. 10, pp. 3007-3017, Dec., 1982

A.J. Viterbi, "A Processing Satellite Transponder for Multiple Access by Low-Rate Mobile Users", Proc. Digital Satellite Communications Conference, Montreal, October, 1978, pp. 166-174

O. Yue, "Spread Spectrum Mobile Radio, 1977-1982", IEEE Trans. on Veh. Tech., Vol. VT-32, No. 1, pp. 98-105, Feb., 1983